RBFS User Guides

Version 25.1.1.1, 18 June 2025

Table of Contents

1. Overview Guides

1.1. RBFS CLI Overview

1.1.1. RBFS Command Line Interface

RBFS CLI Overview

RBFS command line interface is a primary user interface that enables you to interact with RBFS for monitoring, configuring, debugging, and maintaining the system. RBFS command line interface, that runs on top of the Ubuntu shell, provides a rich set of commands which allow you to execute various operations on the system.

RBFS CLI commands are organized in hierarchies based on their functionalities. Commands, which are used to execute the same type of functions, have the same hierarchy. For example, to display information, you can use commands that start with 'show'. Delete command, in RBFS, is used to remove an existing configuration.

The RBFS command-line interface has three modes: Configuration mode, Operation mode, and Debug mode.

Operational mode: This is the default mode of RBFS command line interface. Operational mode allows you to execute the operational commands such as show commands to view or monitor various system configuration and its current state.

Configuration mode: Configuration mode allows to execute various configurations for services or features. It also allows you to view the information for the existing configurations.

Debug mode: It allows you to execute troubleshooting or debugging operations in the RBFS system.

Using the CLI

The following are some of the utilities which help you working with the CLI faster and easier.

Complete Partially Typed Commands:

You can press Tab key to complete a partially typed command. It helps you work with commands faster.

Command Options and Description:

If you do not know the options available for a command and the purposes of the options, you can enter the question mark symbol (?). It displays all the available command options and descriptions for that commands.

In any of the modes, if you type the question mark symbol (?), the CLI displays a set of commands which can be executed in that particular mode.

When you execute configurations through CtrlD, and then with the Command Line Interface, it results in error when you commit the configuration through the CLI. The reason is that CtrlD directly interacts with the backend applications (BDS and CONFD) and these changes are not synced with the CLI.

Launching the RBFS CLI

The following example shows how to start the RBFS CLI.

```
supervisor@rtbrick>LEAF01:~$ cli
supervisor@rtbrick>LEAF01: op>
```

'op>' shows you are in operational mode.

CLI Prompt

This section introduces you to three types of prompts:

- **RBFS CLI prompt**: The RBFS CLI prompt is the command-line interface (CLI) used to interact with the RtBrick FullStack (RBFS) software. Upon logging into RBFS, you will see the CLI prompt in Operational State mode. To switch to a different mode, see the Switching CLI Modes section.
- **Container shell prompt**: The container shell prompt refers to the command-line interface that appears when you access a Linux container environment where RBFS is installed.
- **ONL prompt**: The Open Network Linux (ONL) prompt refers to the command-line interface that appears when using the RBFS software after it is installed.

RBFS CLI Prompt

The RBFS CLI prompt reflects the static hostname and host OS hostname. In RBFS, the static hostname is the container name and the dynamic hostname is derived from DHCP.

The format of the RBFS CLI prompt is as follows:

<username> @ <static_hostname> > <hostname.host-os>: <mode>

Option	Description
<username>	This specifies the user-defined name.
static_hostname	This is specified in the /etc/hostname directory and always corresponds to the container name.
hostname.host-os	This is specified in the /etc/hostname.host-os directory and reflects a DHCP-learned hostname.
mode	This identifies the CLI mode, such as cfg (Configuration mode), op (Operation mode (default mode)), or dbg (Debug) mode. To switch between modes, enter the "switch-mode <mode>" command. For details on switching between modes, see the Switching CLI Modes section below.

Switching CLI Modes

RBFS CLI has three modes: Configuration mode, Operation mode, and Debug mode. You can enter switch-mode command to change the CLI mode.

The following example shows the options available in the "switch-mode" command.

```
supervisor@rtbrick>LEAF01: op> switch-mode ?
    config                  Enter a given mode
    debug                   Enter a given mode
    operation               Enter a given mode
```

The following examples shows how to switch between modes and how the CLI prompt changes with each mode.

Switching from operation (op) to configuration (cfg) mode:

```
supervisor@rtbrick>LEAF01: op> switch-mode config
Activating syntax mode : cfg [config]
supervisor@rtbrick>LEAF01: cfg>
```

Switching from configuration (cfg) to operation (op) mode:

```
supervisor@rtbrick>LEAF01: cfg> switch-mode operation
Activating syntax mode : op [operation]
supervisor@rtbrick>LEAF01: op>
```

Switching from configuration (operation) to debug (debug) mode:

```
supervisor@rtbrick>LEAF01: op> switch-mode debug
Activating syntax mode : dbg [debug]
supervisor@rtbrick>LEAF01: dbg>
```

Container Shell Prompt

The container shell prompt reflects the host OS’s static hostname and hostname.

The format of the container shell prompt is as follows:

<username> @ <static_hostname> > <hostname.host-os>:<current_directory> $

Option	Description
<username>	This specifies the user-defined name.
static_hostname	This is specified in the /etc/hostname directory and always corresponds to the container name.
hostname.host-os	This is specified in the /etc/hostname.host-os directory and reflects a DHCP-learned hostname.
<directory>	This specifies your current working directory.

Example:

```
rtbuser@rtbrick>LEAF01:~ $ cat /etc/hostname
rtbrick

rtbuser@rtbrick>LEAF01:~ $ cat /etc/hostname.host-os
LEAF01
```

ONL Prompt

You can log in into Open Network Linux (ONL) using SSH port 1022, which applies to both in-band and out-of-band access.

The format is as follows:

ssh <user>@<host-ip> -p 1022

Option	Description
<user>	This is the unique identifier you assign to a user account while installing RBFS.
<host-ip>	This specifies the host IP address.

```
ssh rtbuser@192.0.2.0
rtbuser@onl:~ $
```

Turning Paging On/Off

To turn the paging on or off, use the following command:

paging [**on** | **off**]

- off - Pagination will be turned off for the commands that span more than screen length
- on - Pagination will be turned on for the commands that span more than screen length

Example:

```
supervisor@rtbrick>LEAF01: cfg>  paging on
```

Turning On or Off CLI Colors

To enable or disable the colors in the CLI output, use the following command:

color [**on** | **off**]

Example:

```
supervisor@rtbrick>LEAF01: cfg> color on
```

Accessing CLI Logs

RBFS supports sending command history log messages to Graylog, a log management software that enables real-time analysis of log messages.

The command history logs help you to understand which user has executed a specific command across multiple CLI sessions.

The log format for CLI command history logs is: *User '%s' executed command '%s'*.

System logging is implemented for RESTCONF.

For RESTCONF error logs, do not set the log level to 'info'. If you set the log level to info, logs are generated for all the restconfd requests.

Basic CLI Features

Viewing CLI Tree

The show cli tree command displays all RBFS CLI commands and command options that exist.

- show cli tree: This command provides you the list of all RBFS CLI commands and options that are available.
- show cli tree detail: The detail option for the command displays additional information such as minimum and maximum values allowed.

Viewing Command History

The history command enables you to view the previously executed commands. You can execute the command in any of the CLI modes.

history

Example:

```
supervisor@rtbrick>LEAF01: op> history
show config set
```

```
exit
show config set
load config test.json
load config obj.json
show config set
exit
show config set
load config test.json
switch-mode config
load config test.json
load config obj.json
exit
switch-mode config
show config set
load config test.josn
load config obj
load config obj.json
exit
show config set
load config obj.json
load config test.json
exit
show bd running-status
load config test.json
show config set
exit
show bd running-status
exi
show datastore confd table test index index2
exit
```

Setting Configuration Parameters

The set command is used to configure parameters.

```
supervisor@rtbrick: cfg> set system ntp server MyNTPServer ipv4-address 10.1.1.100
```

Deleting Configuration Parameters

The delete command is used to remove existing configuration parameters.

```
supervisor@rtbrick: cfg> delete system ntp server MyNTPServer ipv4-address
10.1.1.100
```

Viewing Core Dump Files

When a router crashes, it is useful to obtain a complete copy of the memory image, also known as core dump, to identify the cause of the crash.

The show core-dump stack-trace command displays core dump data generated by the system. It includes information such as Core Dump File Name, Process Name/ID, Time of Core Dump, Core Dump Size, Signal Information, Process State, and so on.

You can use the command in the following ways:

- show core-dump lists the core dump files.
- show core-dump stack-trace displays all the stack traces (list of the function calls) for all valid core files.
- show core-dump stack-trace <file-name> displays the stack traces for a specified file.
- show core-dump stack-trace <file-name> per-thread displays stack trace for all threads for the specified file.
- Use the clear core-dump to clear output populated.

Viewing Hardware Resource Usage Limit Information

In RBFS, you can view the hardware resource usage limit details.

Run the following command:

show hardware limits

Example:

```
supervisor@rtbrick>rtbrick.net: op> show hardware limits
Hardware resources:
  Module: fib
    ASIC                             : q2c
    Role                             : accessleaf
    Model                            : agcva48s
    IPv4 route count                 : 1200000
    IPv6 route count                 : 250000
  Module: fib
    ASIC                             : q2c
    Role                             : accessleaf
    Model                            : as7946-74xkb
    IPv4 route count                 : 1200000
    IPv6 route count                 : 250000
  Module: fib
    ASIC                             : q2c
    Role                             : accessleaf
    Model                            : as7946-30xb
    IPv4 route count                 : 1200000
    IPv6 route count                 : 250000
  Module: fib
    ASIC                             : q2c
    Role                                : accessleaf
    Model                               : s9600-72xc
    IPv4 route count                    : 1200000
    IPv6 route count                    : 250000
  Module: bgp
    ASIC                                : q2c
    6PE label                           : 2
  Module: confd
    ASIC                                : q2c
    Max MTU profile                     : 8
    Max L3 MTU profile                  : 3
    Max subscriber MTU profile          : 6
    Max physical MTU profile            : 8
  Module: rd
    ASIC                                : q2c
```

Rebooting Containers and Hosts

The reboot command allows you to restart containers and hosts.

reboot <option>

Option	Description
-	Without any option, this command allows you to reboot a container (default). You are prompted to confirm rebooting the container when you enter this command. You must answer yes or no.
container	This command allows you to reboot a container. You are prompted to confirm rebooting the container when you enter this command. You must answer yes or no.
container-and-confirm	This command reboots the container without prompting yes/no.
device	This command allows you to reboot a device. You are prompted to confirm rebooting the device when you enter this command. You must answer yes or no.
device-and-confirm	This command reboots the device without prompting yes/no.

Example:

```
supervisor@rtbrick>LEAF01: cfg> reboot container
```

Viewing System Version Details

To display the version details of RBFS and its various components, use the show version command.

show version

Example:

```
supervisor@ixr_pe1>srv3.nbg1.rtbrick.net: op> show version
UUID         : 2abb4250-2a14-4e5c-84e2-6785eee158f8
Version      : 22.6.0-g4internal.20220620060710+Bfs0000bgpauthlatest.C3abc099d
Role         : spine
Platform     : virtual
Format       : lxd
Build date   : 2022-06-20 06:07:10 UTC
Component                          Version
Timestamp                   Branch
alertmanager                       0.20.1001-
internal.20220613124702+Bdevelopment....       2022-06-07 20:01:29
development
cligen                             0.1.0-
internal.20220613140225+Bdevelopment.C9457c97b     2022-06-07 20:00:33
development
clixon                             4.3.1-
internal.20220618124913+Bdevelopment.C85593b60     2022-06-13 11:48:32
development
<...>
```

Viewing Date and Time

To display system date and time, use the date command.

date

Example:

```
supervisor@rtbrick>LEAF01: op> date
Thu Apr 28 09:56:32 UTC 2022
```

Viewing the Running configuration

To display the running configurations, enter the show config command.

Example:

```
supervisor@rtbrick>LEAF01: cfg> show config
```

Viewing a Specific Hierarchy

To display the configurations under a specific hierarchy, enter the show config <hierarchy> command.

```
supervisor@rtbrick: cfg> show config ?
  access                  Global access configuration
  daemon-options          List of daemon options
  global                  Global configurations
  instance                Network instance configuration
  <...>
```

Viewing Configurations in a Specific Format

The show config command displays the current committed configurations of the system. By default, this command displays the configurations in a json format.

show config <format>

You can also specify the format explicitly, if needed. The available display formats are:

- **json**: Display configurations in JSON format
- **set**: Display configurations in CLI format (similar to commands executed)
- **netconf**: Display configurations in XML format
- **text**: Display configurations in textual format (similar to YANG definition)

The following example shows how configurations are displayed in the text format.

```
supervisor@rtbrick>LEAF01: op> show config text
daemon-options {
    instance-name *;
    afi *;
    safi *;
    bd-type bgp.appd;
    bd-name bgp.appd.1;
}
daemon-options {
    instance-name *;
    afi *;
    safi *;
    bd-type bgp.iod;
    bd-name bgp.iod.1;
}
interface {
    name lo-0/0/0;
    unit {
        unit-id 0;
        address {
            ipv4 {
                prefix4 198.51.100.75/24;
            }
            ipv6 {
                prefix6 2001:db8:0:110::/32;
            }
        }
    }
}
<...>
```

To view configurations in the set format, use the show config set command.

Example:

```
supervisor@rtbrick>LEAF01: cfg>  show config set
set interface ifp-0/0/1
set interface ifp-0/0/1 ifp-id 1
set interface ifp-0/0/2
set interface ifp-0/0/2 ifp-id 2
set instance blue
set instance blue protocol bgp address-family ipv4 multicast
set instance blue protocol bgp address-family ipv6 unicast
set instance red
set instance red protocol bgp address-family ipv4 unicast
set instance red protocol bgp address-family ipv6 unicast
```

Viewing Configuration in a Specified Hierarchy

To view configuration in a specified hierarchy, use the following command:

```
supervisor@rtbrick>LEAF01: cfg> show config set instance red protocol bgp
set instance red protocol bgp address-family ipv4 unicast
set instance red protocol bgp address-family ipv6 unicast
```

Committing CLI Configurations

To apply the configurations, use the commit command.

The following example shows how to commit your changes using the commit command.

```
supervisor@rtbrick>LEAF01:~$ cli
supervisor@rtbrick>LEAF01: op> switch-mode config
supervisor@rtbrick>LEAF01: cfg> <cli command goes here>
supervisor@rtbrick>LEAF01: cfg> commit
```

When you exit CLI configuration with uncommitted changes, a reminder text appears saying that you have changes to commit, as shown in the following example:

```
supervisor@rtbrick>LEAF01: cfg> exit
Uncommitted changes are present
1. Discard the changes and exit
2. Commit the changes and exit
3. Keep the changes and exit [Default behavior]
Enter one of the above choice to proceed :
```

Viewing Uncommitted Changes

To view the uncommitted changes, use the show diff command. The show diff command displays output in JSON format, whereas the the show diff set command displays the output in CLI format.

```
supervisor@rtbrick>LEAF01: cfg> show diff

supervisor@rtbrick>LEAF01: cfg> set interface ifp-0/0/3 ifp-id 3
supervisor@rtbrick>LEAF01: cfg> set interface ifp-0/0/4 ifp-id 4
supervisor@rtbrick>LEAF01: cfg> show diff set
+set interface ifp-0/0/3
+set interface ifp-0/0/3 ifp-id 3
+set interface ifp-0/0/4
+set interface ifp-0/0/4 ifp-id 4
supervisor@rtbrick>LEAF01: cfg> show diff
 }
+interface {
+    name ifp-0/0/3;
+    ifp-id 3;
+}
+interface {
+    name ifp-0/0/4;
+    ifp-id 4;
+}
 instance {
```

Adding a Configuration Description

An in-line description or comment can be added to a system configuration to describe it.

set system config-description <description>

Example:

```
supervisor@rtbrick>LEAF01: cfg> set system config-description "This is sample test
```

```
configuration"
supervisor@rtbrick>LEAF01: cfg> commit
supervisor@rtbrick>LEAF01: cfg> show config
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "config-description": "This is sample test configuration"
    }
  }
```

Saving Configuration

To save configurations, enter the following command:

```
supervisor@rtbrick>LEAF01: cfg> save config my_config.json
```

- Ensure that you use .json at the end of the filename.
- The configuration will be saved to the current working directory of CLI executable.

Deleting the Entire Running Configuration

The discard command is used to discard configuration changes. To delete the entire running configurations (both committed and uncommitted configurations) at a time, use the discard all command.

Example:

```
supervisor@rtbrick>LEAF01: cfg> discard all
```

Discarding the Uncommitted Configuration

To discard the uncommitted configuration, enter the following command:

```
supervisor@rtbrick>LEAF01: cfg> discard
```

Viewing the Configuration Differences in the Current and Previous Versions

In RBFS, you can view the configuration differences between the current and the previous versions.

show diff <number>

```
supervisor@rtbrick>LEAF01: cfg> show diff 2
 system {
-    secure-management-status false;
+    secure-management-status true;
 }
```

Rolling Back to a Previously Committed Configuration

To rollback to a specific configuration prior to the most recently committed one, use the following command:

rollback <number>

number: Specifies the rollback ID. Range: 1 through 49. 0 refers to the active configuration, 1 refers to the most recent previous configuration. Default: 1

For example, to rollback to rollback ID 2, use the following command:

```
supervisor@rtbrick>LEAF01: cfg> rollback 2
```

Rolling Back to a Specific Version of Software Configuration

To rollback to a specific version of the software configuration, use the following command:

rollback commit-id <commit-hash>

Example:

```
supervisor@rtbrick>LEAF01: cfg> rollback commit-id
29d5db038c1920fdsdsdsdsdsd323232
```

Loading Configuration

To load configurations, enter the following command:

load config <filename> <option>

The options include merge and replace. You can specify merge after the file name to merge the configuration with the running configuration. Specify replace to replace the running configuration with the new one. Without any option, it replaces the running configuration, by default.

```
supervisor@rtbrick>LEAF01: cfg>  load config <filename>
```

- Ensure that you use '.json' at the end of the filename.
- Remember to commit your changes after loading.

Viewing Routes

The show route command displays information of routes.

Syntax:

show route <options>

Attribute	Description
-	Without any option, the command displays the information for all routes for all modules.
detail	Shows detailed route information.
instance <name>	Routing table information for a specified instance.
ipv4	Shows route information for the IPv4 routing table.
ipv6	Shows route information for the IPv6 routing table.
mpls	Shows route information for the MPLS routing table.
label <value>	Shows route information for a specified destination label.
prefix <value>	Shows route information for a specified destination prefix.
prefix-length-distribution	Shows the number of routes with the same prefix length for the sources.
source	Shows routes from a specified source.
summary	Shows the number of routes selected by RIBD for each source.

Example 1: Route information

```
supervisor@rtbrick>LEAF01: op> show route
Instance: default, AFI: ipv4, SAFI: unicast
Prefix/Label        Source          Pref    Next Hop                Interface
11.0.0.1/32         arp-nd          6       11.0.0.1                hostif-0/0/4/1
```

```
12.1.0.0/24         ospf              10      23.0.0.2                hostif-0/0/0/1
23.0.0.0/24         direct            0       23.0.0.0                hostif-0/0/0/1
25.0.1.0/24         ospf              10      23.0.0.2                hostif-0/0/0/1
25.1.1.0/24         ospf              10      23.0.0.2                hostif-0/0/0/1
34.0.3.3/32         direct            0       34.0.3.3                hostif-0/0/2/1
56.0.1.4/30         ospf              10      23.0.0.2                hostif-0/0/0/1
56.0.2.0/31         ospf              10      34.0.2.4                hostif-0/0/1/1
```

Example 2: Route summary

```
supervisor@rtbrick>LEAF01: cfg> show route summary
Instance: default
  Source                 Routes
  bgp                         2
  direct                      4
  Total Routes                6
Instance: ip2vrf
  Source                 Routes
  bgp                         6
  direct                      2
  Total Routes                8
Instance: li-vrf
  Source                 Routes
  bgp                         4
  direct                      2
  Total Routes                6
Instance: mgmt-vrf
  Source                 Routes
  bgp                         2
  direct                      2
  Total Routes                4
Instance: radius-vrf
  Source                 Routes
  bgp                         5
  direct                      2
  Total Routes                7
```

Example 3: Routes with the same prefix length for IPv4

```
supervisor@rtbrick>LEAF01: cfg> show route prefix-length-distribution
Instance: default
  Prefix Length      Count
            /32          2
           /128          4
            Sum          6
Instance: ip2vrf
  Prefix Length      Count
             /0          2
            /24          1
            /32          2
            /64          1
           /128          2
            Sum          8
Instance: li-vrf
  Prefix Length      Count
             /0          2
            /32          2
           /128          2
            Sum          6
Instance: mgmt-vrf
  Prefix Length      Count
            /32          2
           /128          2
            Sum          4
Instance: radius-vrf
  Prefix Length      Count
             /0          2
            /24          1
            /32          2
           /128          2
            Sum          7
```

Viewing Route Resolution

The show route-resolution command displays the routes which were requested to be resolved for their nexthops. Otherwise, it shows the route is unresolved. **Syntax**:

show route-resolution <options>

-	**Without any option, the command displays the information for all requests and response tables side by side.**
destination-instance	Displays the information for all requests and response for a destination instance.
look-up instance	Displays lookup instance routes.
prefix	Displays routes for prefix 4 or prefix 6.
resolved	Displays resolved routes.
source	Displays source of requested source.
unresolved	Displays unresolved routes.

Example:

```
supervisor@L1-STD-2-2008>bm08-tst.hel.rtbrick.net: op> show route-resolution
192:1::1, Source: bgp
  Destination instance: default, AFI: ipv4, SAFI: vpn-unicast
  Lookup      instance: default, AFI: ipv6, SAFI: labeled-unicast
  Covering Prefix: 192:1::1/128
    Interface        MAC Address         Nexthop
    hostif-0/0/1/10  7a:11:21:c0:00:03   fe80::7811:21ff:fec0:3
192:1::1, Source: bgp
  Destination instance: default, AFI: ipv4, SAFI: vpn-multicast
  Lookup      instance: default, AFI: ipv6, SAFI: labeled-unicast
  Covering Prefix: 192:1::1/128
    Interface        MAC Address         Nexthop
    hostif-0/0/1/10  7a:11:21:c0:00:03   fe80::7811:21ff:fec0:3
    <...>
```

Example:

```
supervisor@rtbrick>ufi07.q2c.u21.r4.nbg.rtbrick.net: cfg> show route-resolution
unresolved
192.168.16.128, Source: radius
  Lookup      instance: inband_mgmt, AFI: ipv4, SAFI: unicast
  Covering Prefix: None, 7 resolution attempts
198.18.73.251, Source: pim
  Lookup      instance: ip2, AFI: ipv4, SAFI: unicast
  Covering Prefix: None, 7 resolution attempts
```

Managing RBFS Services

Brick Daemons

RBFS microservices architecture allows daemons to serve various complementary functions and provide services.

For example, the subscriber daemon (subscriberd) manages the current subscriber state and is responsible for authentication, authorization, and accounting. The ribd daemon is responsible for route selection, next-hop resolution, tunnel selection and recursion.

There are daemons such as CtrlD (Controller) and ApiGwD (API Gateway) which are part of the RBFS ecosystem. These daemons sit in the middle (on the ONL) and manage all the communication between the client and backend services running in the container. The API Gateway (ApiGwD) daemon provides a single point access to expose services running inside of the RBFS container.

RBFS daemons and other dependencies are packaged as an Ubuntu LXC container. The RBFS container is hosted on the Open Network Linux (ONL), an open-source operating system, which can be run on white box switches.

RBFS can perform various roles such as Spine, Leaf, and Multiservice Edge which serve different use-cases. The software images of these various roles contain daemons that are required to serve these roles for their different functions. The RBFS Multiservice Edge software image contains all the RBFS daemons packaged in a container, other roles such as Spine and Leaf include only the daemons which are required to carry out their respective functions.

For example, the Spine RBFS image includes (in addition to other daemons) the interior gateway protocol daemons such as isis.appd, isis.iod, ospf.appd, and ospf.iod which are not required in the Access Leaf image.

Similarly, the Access Leaf image should include daemons (in addition to other daemons) such as subscriberd, l2tpd, pppoed, and ipoed which are not present in the Spine image.

The daemons such as alertmanager, confd, etcd, fibd, hostconfd, ifmd and so on are present in the images of both the Spine and Leaf roles as these daemons are required in both of these roles.

Launching Microservices Dynamically

When the RBFS container starts up, it starts only relevant microservices depending on the image role and platform. This is done to minimize unnecessary resource consumption. In RBFS, there are base microservices and on-demand microservices. RBFS containers will have all microservices installed according to the platform and image role, but not all will be enabled on bootup. Only the base microservices will be enabled and started on bootup. On-demand microservices will only be started when their respective configurations are configured and will stop once all dependent configurations are deleted.

For instance, when the user configures BGP with the CLI command set instance <instance> protocol bgp local-as <as-number>, the rtbrick-bgp.appd.1 and rtbrick-bgp.iod.1 services will start. And, once the BGP configuration is deleted, "rtbrick-bgp.appd.1" and "rtbrick-bgp.iod.1" will be stopped after 5 minutes (graceful shutdown time).

By default, the following base microservices will be running in the container.

- rtbrick-confd
- rtbrick-etcd
- rtbrick-fibd
- rtbrick-hostconfd
- rtbrick-ifmd
- rtbrick-lldpd

- rtbrick-mribd
- rtbrick-opsd
- rtbrick-poold
- rtbrick-resmond
- rtbrick-resmond-agent
- rtbrick-restconfd
- rtbrick-ribd
- rtbrick-staticd

When you make other RBFS configurations, the required on-demand microservices will be automatically enabled.

Listing All Available Services

To list all available RBFS services, enter the following command:

```
sudo systemctl list-unit-files | grep "rtbrick-"
```

Starting or Stopping the Services

To start or stop a specific RBFS service, enter the following command:

```
sudo service <service> start|stop
```

Restarting a Service

To restart a specific RBFS service, enter the following command:

```
sudo service <service> restart
```

Example:

```
supervisor@onl>multiservice-edge.rtbrick.net:~ $ sudo service rtbrick-ctrld
restart
```

- In the unlikely event of a failure in a daemon, the service will restart automatically.

- If a service has restarted too frequently within a short period, it may be blocked by the system to prevent further instability. In such cases, the service can be unblocked by resetting its failure state with the sudo systemctl reset-failed <service> command. This ensures that the service is no longer considered “failed” by the system and can be managed normally again. Including this information in the documentation will help users manage and troubleshoot their RBFS services effectively, ensuring smoother operations and reducing downtime caused by blocked or unstable services.

Verifying the Status of a Service

To verify the status of a RBFS Service, enter the following command:

```
sudo service <service> status
```

For example, to check the status of the rtbrick-pppoed.1 service, enter the following command:

```
supervisor@onl>multiservice-edge.rtbrick.net:~ $ sudo service rtbrick-pppoed.1
status
```

Removing the Committed/Uncommitted Configuration

To discard the both committed and uncommitted configurations, enter the following command:

```
supervisor@rtbrick>LEAF01: cfg> discard all
```

Exeuting the discard all command will remove your committed configurations. Users will receive a warning indicating that it will permanently remove all committed configurations. A confirmation prompt shows the message providing an explicit [yes/no] option for the user to choose before proceeding.

Example:

```
supervisor@rtbrick.net: cfg> discard all
Warning: This action will discard all committed configurations.
```

```
Are you sure you want to proceed? [yes/no] [Default: yes]: no
supervisor@rtbrick.net: cfg> commit
```

Filtering Output using Pipe (|) in Show Commands

RBFS supports output filtering according to the criteria specified by the options using pipe (|) for all show commands.

Only a single-level pipe with option is supported.

- The pipe (|) command is supported only with show commands.
- The supported two options are: grep and count.

Syntax:

- <show command> | grep <search pattern>
- <show command> | count

Example:

```
supervisor@rtbrick.net: op> show version
UUID         : 4a2b63df-305b-4d15-8729-9cf6c57cb628
Version      : 25.1.0-g6daily.20250123075734+Bdevelopment.Cfbee306f
Role         : multiservice-edge
Platform     : q2a
Format       : lxd
Build date   : 2025-01-23 07:57:34 UTC
supervisor@rtbrick.net: op> show version | grep "Format|Role"
Role         : multiservice-edge
Format       : lxd
supervisor@rtbrick.net: op> show version | count
  <cr>
supervisor@rtbrick.net: op> show version | count
6
```

Error Path Information in RESTCONF Configurations

You can see the error path information, which indicates the exact hierarchy of the problematic node within a module in the RESTCONF configurations. It enables troubleshooting easier by providing references to the interface, peer group, instance, or configuration that is causing the problem.

Example:

```
supervisor@testFix>srv02-tst.r1.nbg.rtbrick.net: cfg> show config set time-series
```

```
supervisor@testFix>srv02-tst.r1.nbg.rtbrick.net: cfg> set time-series metric test
supervisor@testFix>srv02-tst.r1.nbg.rtbrick.net: cfg> show diff set
+set time-series metric test
supervisor@testFix>srv02-tst.r1.nbg.rtbrick.net: cfg> commit
[lime]Jan 27 10:07:19: Commit failed. Edit and try again or discard changes: application missing-element
Mandatory variable 'table-name' missing for node 'metric' with path: time-series/metric/ <bad-element>table-
name</bad-element>
```

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

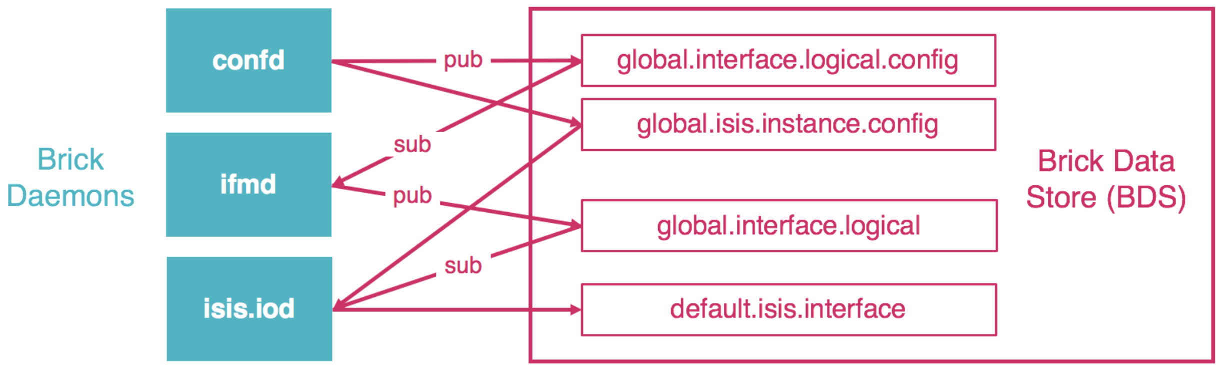
1.2. BDS Overview

1.2.1. BDS Overview

The Brick Data Store (BDS) is a purpose-built, in-memory state database optimized for cloud networking. In RBFS, all system state information is stored as objects in BDS tables. Objects are entries in BDS tables that represent a state.

Pub/Sub Model

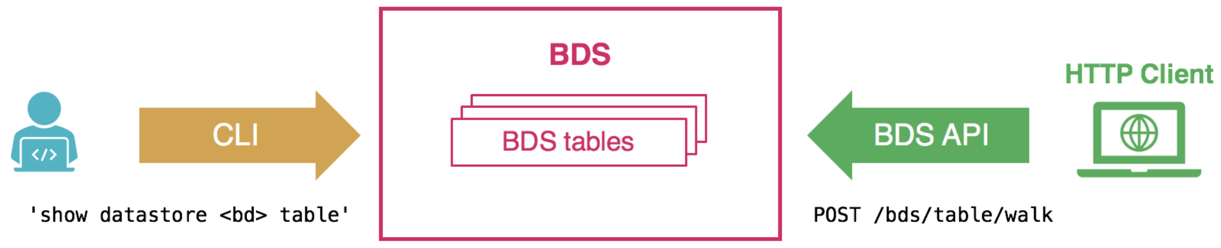
All Brick Daemons (BD) independently publish and subscribe to tables in a pub/sub model. This model provides resilience and scalability. The figure illustrates the concept:



In this example, the configuration daemon (confd) publishes tables that contain configuration data for logical interfaces or IS-IS instances. The interface management daemon (ifmd) is responsible for creating and maintaining interfaces. It therefore subscribes for example to the logical interface configuration table. After processing the data, it creates the logical interfaces and publishes them in the logical interface table. The IS-IS input/output daemon (isis.iod) subscribes to the logical interface table as well as the IS-IS configuration tables. It in turn creates and runs interfaces on which IS-IS protocol packets are exchanged, and publishes them in the IS-IS interface table.

BDS User Interface

All BDS tables and objects are fully accessible to the RBS user both via CLI and an API. This provides unprecedented visibility into the system state. This guide covers the BDS CLI. For the BDS API, refer to the BDS API Reference.



Please note the RBFS CLI supports show commands to verify the configuration and operation of the system for all features. Therefore you usually do not need to inspect BDS tables directly. For example, for verifying the status of the logical interfaces, you can simply use the 'show interface summary' or the 'show interface logical' commands, instead of displaying the logical interface table. The BDS CLI and API to inspect BDS tables are rather available in addition to advanced analysis or troubleshooting.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

1.2.2. BDS Operational Commands

This section summarizes some useful BDS CLI commands. It assumes you have some basic knowledge of BDS, and are familiar with the respective tables you are looking for. Describing all tables involved in a particular feature or functionality is out of the scope of this guide.

BDS Summary

The BDS summary command provides some metadata of the BDS tables.

Syntax:

show datastore <bd-name> **summary** <option>

Option	Description
<bd-name>	Name of the brick daemon to request this information from. As all BDs independently publish and subscribe to BDS tables, they all hold a different set of tables. As a best practice, select the BD that owns the respective table you are looking for.
table <table-name>	Display metadata for the given table.

Example:

```
supervisor@rtbrick>LEAF01: op> show datastore ribd summary
Brick Datastore Summary:
Table Name: local.bds.table.registry.ribd
  Index                                               Type                 Active
Obj Memory   Index Memory
  sequence-index                                      bds_rtb_bplus           234
24.38 KB     9.59 KB
  gc-index                                            bds_rtb_bplus             0   0
bytes      0 bytes
  table-name-index                                    bplus                   234
24.38 KB     9.59 KB
Table Name: local.trim.qrunner.table
  Index                                               Type                 Active
Obj Memory   Index Memory
  sequence-index                                      bds_rtb_bplus             7
840 bytes    728 bytes
  gc-index                                            bds_rtb_bplus             0   0
bytes      0 bytes
  immutable_index                                     bplus                     7
840 bytes    728 bytes
  qrunner-index                                       qrunner                   7
840 bytes    728 bytes
Table Name: local.bds.statistics
  Index                                               Type                 Active
Obj Memory   Index Memory
  sequence-index                                      bds_rtb_bplus           319
34.84 KB     14.36 KB
  gc-index                                            bds_rtb_bplus             0   0
bytes      0 bytes
  immutable-index                                     bplus                   319
34.84 KB     14.36 KB
Table Name: local.bds.module.registry
  Index                                               Type                 Active
```

```
Obj Memory   Index Memory
  sequence-index                                   bds_rtb_bplus           68
3.53 KB      2.79 KB
  gc-index                                         bds_rtb_bplus            0   0
bytes      0 bytes
  module-name-index                                bplus                   68
3.53 KB      2.79 KB
<...>
```

BDS Tables

You can use the BDS table commands to display the table objects that contain the actual state information.

Syntax:

show datastore <bd-name> **table** <option>

Option	Description
<bd-name>	Name of the brick daemon to request this information from. As all BDs independently publish and subscribe to BDS tables, they all hold a different set of tables. As a best practice, select the BD that owns the respective table you are looking for.
<table-name>	Name of the BDS table to display. Without further options, this command displays all objects in a table format.
<table-name> json	Display the complete table data in JSON format.
<table-name> attribute <attribute-name> <attribute-value> exact	Filter the table objects based on attribute name and value. You can filter on any attribute, except for attributes of type array. The filter performs a regex match. You can therefore specify the attribute value as a regular expression (regex). You can use the exact match along with the (default) regular expression match.

Option	Description
<table-name> summary	Display metadata for the given table.
properties	Display owner, published/subscribed, and locality information for all tables known by the given daemon.

Example 1: Logical Interface Table

```
supervisor@rtbrick>LEAF01: op> show datastore ifmd table global.interface.logical
Object: 0, Sequence 100125, Last update: Mon Apr 03 13:49:39 GMT +0000 2023
  Attribute                                 Type                          Length
Value
  logical_unit_id (1)                       uint16 (3)                         2
0
  ifl_name (2)                              string (9)                        13
ifl-0/1/31/0
  ifp_name (3)                              string (9)                        11
ifp-0/1/31
  instance (5)                              string (9)                         8
default
  mac_address (8)                           macaddr (22)                       6
e8:c5:7a:8f:76:f2
  ipv4_status (10)                          uint8 (2)                          1
up
  ipv6_status (12)                          uint8 (2)                          1
up
  mpls_mtu (13)                             uint16 (3)                         2
1500
  mpls_status (14)                          uint8 (2)                          1
up
  iso_mtu (15)                              uint16 (3)                         2
1500
  iso_status (16)                           uint8 (2)                          1
down
  admin_status (17)                         uint8 (2)                          1
up
  link_status (18)                          uint8 (2)                          1
up
  ifl_type (19)                             uint8 (2)                          1
Logical Sub interface
  operational_status (24)                   uint8 (2)                          1
up
  ifindex (25)                              uint32 (4)                         4
63745
  instance_id (27)                          uint32 (4)                         4
0
<...>
```

Example 2: Filter IPv6 Route Table by Prefix

```
supervisor@rtbrick>LEAF01: op> show datastore ribd table default.ribd.1.fib-
local.ipv6.unicast
 attribute prefix6 2001:db8::1/128
```

```
Object: 0, Sequence 1900002, Last update: Mon Apr 03 13:49:39 GMT +0000 2023
  Attribute                                 Type                           Length
Value
  prefix6 (4)                               ipv6prefix (16)                    17
2001:db8::1/128
  source (11)                               uint8 (2)                           1
direct
  sub_src (12)                              uint8 (2)                           1
Host
  nexthop_key (25)                          payload (8)                        24
3c86eaebe6617cf61ed96c819cfd63839bd90cc85a533067
  preference (40)                           uint32 (4)                          4
0
  bcm_status (52)                           uint8 (2)                           1
None
  return_code (53)                          uint32 (4)                          4
0
  vpp_status (54)                           uint8 (2)                           1
None
  route_status (55)                         uint32 (4)                          4
|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-|-
```

Example 3: Filter IPv6 Route Table with Exact Match

```
supervisor@rtbrick>LEAF01: op> show datastore bgp.appd.1 table ip2vrf.bgp.rib-
in.ipv4.unicast.198.51.100.30.198.51.100.25 attribute prefix4 198.51.100.11/24
exact
Object: 0, Sequence: 367772, Last update: Wed May 19 08:05:08 GMT +0000 2021
  Attribute                                 Type                           Length
Value
  status (1)                                uint8 (2)                           1
Valid
  recv_path_id (2)                          uint32 (4)                          4
0
  prefix4 (3)                               ipv4prefix (13)                     5
198.51.100.40/24
  rd (5)                                    route-distinguisher (40)            8
198.51.100.100:65001
  source (6)                                uint8 (2)                           1
bgp
  sub_src (7)                               uint8 (2)                           1
Local-Peer
  as_path (9)                               array (7), uint32 (4)              20
[57381, 42708, 1299, 5511, 3215]
  origin (10)                               uint8 (2)                           1
IGP
  peer_type (12)                            uint8 (2)                           1
2
  igp_metric (13)                           uint32 (4)                          4
4294967295
  send_path_id (18)                         uint32 (4)                          4
3238151775
  bgp_nh4 (19)                              ipv4addr (12)                       4
198.51.100.30
  community (24)                            array (7), community (27)           8
['1299:20000', '42708:200']
```

BDS Schema

The Brick Data Store is schema-driven. Table and object schema definitions are located in RBFS in /usr/share/rtbrick/libbds/. Instead of inspecting schema files, you can use the BDS schema commands to view the schemata directly in the CLI.

Syntax:

show datastore <bd-name> **schema** <option>

Option	Description
<bd-name>	Name of the brick daemon to request this information from. As all BDs independently publish and subscribe to BDS tables, they all hold a different set of tables. To view a table or object schema, you can select any BDs that know the respective table.
table-name <table-name>	Display the schema of the given table.
object object-name <object-name>	Display the schema of the given object.
object table-name <table-name>	Display the schema of the object for a given table. This option is useful if you do not know the name of the object but the name of the table in which it is used.

BDS Statistics Memory

The BDS statistics memory command provides detailed memory usage information.

Syntax:

show datastore <bd-name> **statistics memory**

Option	Description
<bd-name>	Name of the brick daemon of which to display the memory usage information.

2. Routing

2.1. RIB

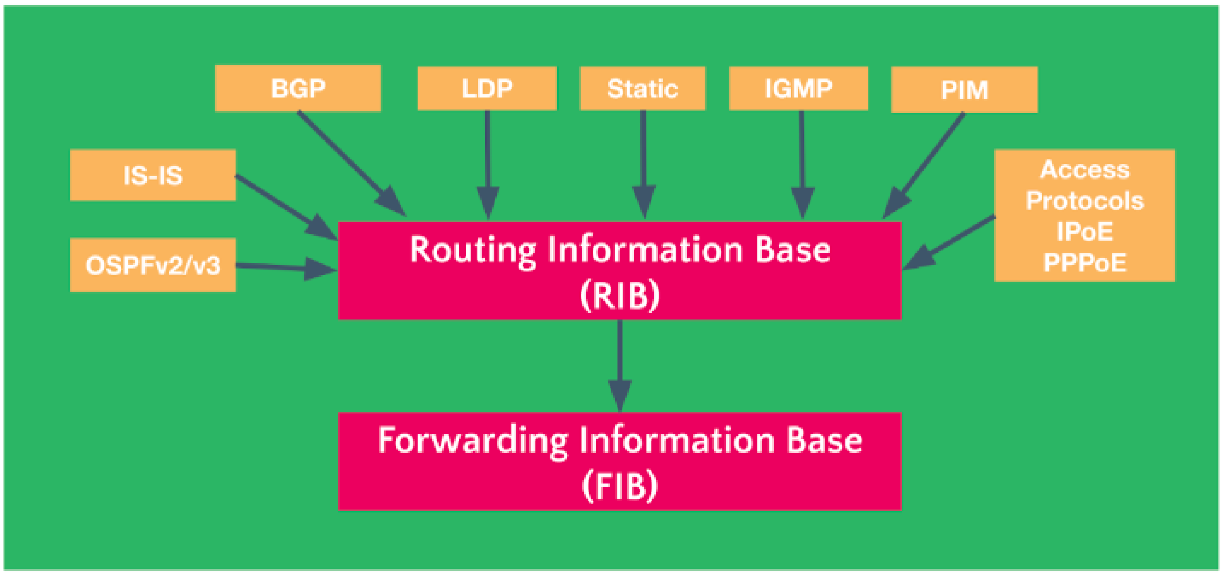
2.1.1. RIB Overview

A Routing Information Base (RIB) stores and maintains the route information. The RIB table contains information about each reachable network prefix and the next hop information. Each routing protocol inserts its routes into the RIB when it learns a new route. The RBFS route manager (ribd) selects the best routes from the protocol RIBs and places them into the Forwarding Information Base (FIB) table. If a destination becomes unreachable, the route is marked unusable and eventually removed from the RIB

The main differences between RIB and FIB are:

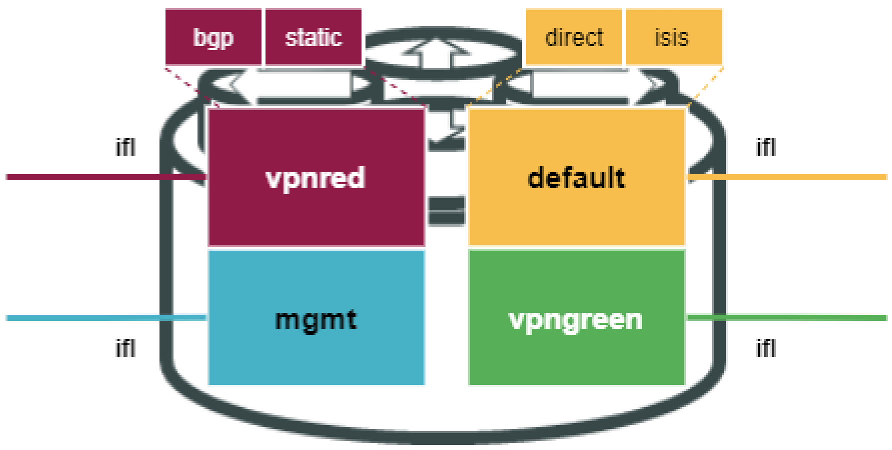
- RIB is about storing and maintaining route information, including attributes for route selection (control plane), etc.
- FIB is used to determine how packets are forwarded (data plane), i.e., only contains a subset of information from RIB. Also, FIB is downloaded to ASIC or VPP.

The image below shows how various routing protocols insert their routes into the RIB.


BGP
LDP
Static
IGMP
PIM
IS-IS
OSPFv2/v3
Access Protocols IPoE PPPoE
Routing Information Base (RIB)
Forwarding Information Base (FIB)

Routing Instances

A routing instance represents a set of interfaces, routing protocols, and corresponding routing tables. You can think of a routing instance as a virtual router within RBFS.



Each logical interface (ifl) is associated uniquely with a routing instance. If no routing instance is explicitly configured, the default routing instance is used. The interface association can be seen with the show interface address command.

Routing instances are configured using the set instance <instance_name> syntax. When configuring a routing instance, you must specify which address families (AFI, SAF) the routing instance should support.

Route Preference

Several routing protocols, including static routes, may provide multiple paths to reach a particular destination. However, not all of these paths are necessarily optimal. At any given time, only one routing protocol can identify the most optimal route to the destination. Each routing protocol, including static routes, is given a preference value, where lower values indicate higher preference. In cases where multiple routing sources are present, the route identified by the routing protocol with the highest preference is selected as the best option and added to the local routing table.

Routing Protocol	Route Preference
Direct	0
Static	2
IPoE	7

Routing Protocol	Route Preference
PPP	8
LDP	9
OSPF	10
IS-IS LEVEL 1	15
IS-IS LEVEL 2	18
eBGP	20
iBGP	200

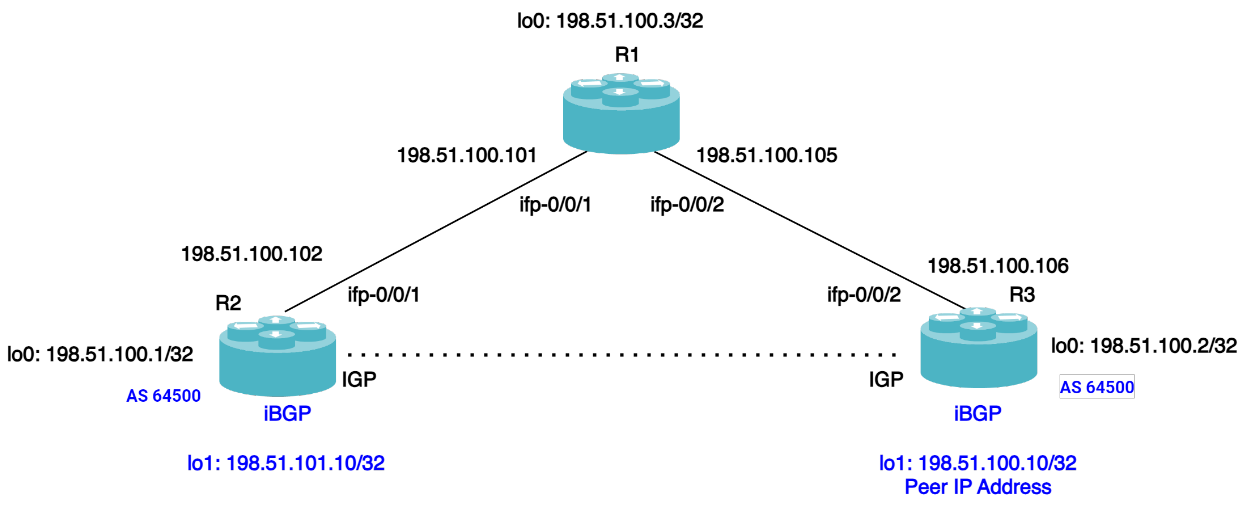
Route Selection

RIBD obtains routes and corresponding next-hops from various sources such as static, protocols, and direct connections. Route selection and resolution are also handled by RIBD, which chooses the best route based on the preference assigned to each source. If multiple sources provide the same prefix, RIBD will select the best route based on the source’s preference and install it in FIBD.

Route Resolution

Route resolution is the process of finding the forwarding next hop from protocol nexthop downloaded into RIBD. For this purpose RIBD performs recursive route lookups till it finds a direct outgoing interface for the route.

For example, BGP installs a route with nexthop set to its peer IP address. However, it doesn’t provide the outgoing interface. So, RIBD will check the peer IP address (next-hop IP) in its routing table (most probably downloaded by IGPs) and find the directly connected router’s IP address and its outgoing nexthop.



The BGP route for 198.51.100.10/32 will only be added to the routing table of router R2 if the IP address listed as the next-hop attribute is already reachable based on the information stored in the routing table. Additionally, the BGP route that is installed will contain a reference to the next-hop address 198.51.100.2 (refer to the figure above).

The network 198.51.100.10/32 can be accessed through an IP address that is not directly connected. The physical interface is not located where the BGP route is installed, so it is added to the IP routing table without any information about the outgoing interface. To find the BGP next-hop in the routing table, the router must perform a recursive lookup. However, to use a BGP route, the BGP next-hop IP address must be reachable. This is usually done through an IGP that provides reachability information. In this case, the BGP next-hop 198.51.100.2 is found in the routing table of R2, which is known via OSPF. The outgoing interface is ifp-0/0/1.

During the first route lookup, the router checks whether the destination prefix is in the routing table. If it is, then a recursive lookup is performed for the next-hop IP address. Since the next hop address is not a directly connected interface, the router needs to do a recursive lookup to find it.

Below is an example of the BGP route resolution for prefix:198.51.100.10/32:

```
supervisor@rtbrick.net: op> show route-resolution resolved prefix 198.51.100.10
198.51.100.10, Source: bgp
  Destination instance: default, AFI: ipv4, SAFI: unicast
  Lookup      instance: default, AFI: ipv4, SAFI: unicast
  Covering Prefix: 198.51.100.10
    Interface         MAC Address         Nexthop
    ifl-0/0/1/13     e8:c5:7a:8f:56:47    198.51.100.101
```

Policy Attachments

Routing Policies are a set of rules that enable you to manage and alter the default behavior of routing protocols, like BGP and IS-IS. Such policies consist of various "terms," which include "match" and "action" sections with control. The traffic that matches the "match" block is handled by the "action" block.

Once policies have been created, they need to be applied to take effect. Attachment points describe the specific applications and processes to which policies can be applied.

For more information, see section “2.2.4. Attaching Policies” of the Policy User Guide.

Nexthops

Nexthop helps routers determine the best path for forwarding data packets to their final destination efficiently. The next hop is identified by its IP address, which is stored in the routing table alongside the associated network prefix and other routing information.

Adjacency

The FIB learns the routing information from the routing table and tracks the next hop for all routes. The adjacency table maintains Layer 2 information(Nexthop) for the routes listed in the FIB (the resolved routes that can be installed into the hardware).

FIB

FIB determines how packets are forwarded (data plane), i.e., it only contains a subset of information from RIB. Also, FIB is downloaded to ASIC or VPP.

The main difference between the RIB and the FIB is that the RIB contains all the routes the router has learned, while the FIB only contains the best paths to each destination network. The RIB is constantly updated by the routing protocols and static configuration, while the FIB is only updated when the router needs to recalculate the best paths. The FIB is also updated more quickly than the RIB since it only needs to store the best paths and not all routes. The FIB maintains next-hop address information based on the information in the IP routing table.

2.1.2. RIB Operational commands

Display Routes

The show route command displays information on routes.

Syntax:

show route <option>

Option	Description
-	Without any option, the command shows the information for all routes for all instances.
detail	Shows detailed route information.
import	The commands show all the routes imported from another instance. For information about route import command options, see section Show Route Import.
incoming interface <name>	Shows the information of the specified interface through which the routes enter to the router.
instance <name>	Shows routing table information for a specified instance.
hierarchy	Shows routing information in a hierarchical
ipv4	Shows route information for the IPv4 routing table.
ipv6	Shows route information for the IPv6 routing table.
l2vpn	Shows route information for the L2VPN routing table. For information about l2vpn command options, see section Show Route L2VPN.
mpls	Shows route information for the MPLS routing table.
prefix <value>	Shows route information for a specified destination prefix.
prefix-length-distribution	Shows the number of routes with the same prefix length for the sources.
service	Shows the number of routes with the same service label.
source	Shows routes from a specified source.

Option	Description
summary	Shows the number of routes selected by RIB for each source.

Example 1: The following example shows route information.

```
supervisor@rtbrick>LEAF01: op> show route
Instance: default, AFI: ipv4, SAFI: unicast
Prefix/Label                             Source            Pref    Next Hop
Interface
192.1.0.3/32                             direct            0       192.1.0.3
lo-0/0/0/1

Instance: default, AFI: ipv4, SAFI: labeled-unicast
Prefix/Label                             Source            Pref    Next Hop
Interface                      Label
192.1.0.3/32                             direct            0       192.1.0.3
lo-0/0/0/1                     -
Instance: default, AFI: ipv6, SAFI: unicast
Prefix/Label                             Source            Pref    Next Hop
Interface

192:1::2/128                             bgp               20      fe80::eac5:7aff:fe8f:5663
ifl-0/1/70/13
192:1::3/128                             direct            0       192:1::3
lo-0/0/0/1

192:1::4/128                             bgp               20      fe80::eac5:7aff:fe8f:5663
ifl-0/1/70/13
Instance: default, AFI: ipv6, SAFI: labeled-unicast
Prefix/Label                             Source            Pref    Next Hop
Interface                      Label
192:1::2/128                             bgp               20      fe80::eac5:7aff:fe8f:5663
ifl-0/1/70/13                  -
192:1::3/128                             direct            0       192:1::3
lo-0/0/0/1                     -
192:1::4/128                             bgp               20      fe80::eac5:7aff:fe8f:5663
ifl-0/1/70/13                  2020
```

Example 2: A route summary is shown in the following example.

```
supervisor@rtbrick>LEAF01: op>  show route summary
Instance: default
  Source                 Routes
  bgp                         4
  direct                      4
  Total Routes                8
Instance: inband-vrf
  Source                 Routes
  bgp                         4
  direct                      2
  Total Routes                6
Instance: ip2vrf
  Source                 Routes
  bgp                         3
  direct                      2
  Total Routes                5
Instance: li-vrf
  Source                 Routes
  bgp                         2
  direct                      2
  Total Routes                4
Instance: mgmt-vrf
  Source                 Routes
  bgp                         4
  direct                      2
  Total Routes                6
Instance: radius-vrf
  Source                 Routes
  bgp                         3
  direct                      2
  Total Routes                5
```

Example 3: This example shows routes with IPv4 prefixes of the same length

```
supervisor@rtbrick>LEAF01: op>> show route prefix-length-distribution
Instance: default
  Prefix Length      Count
            /32          2
           /128          6
            Sum          8
Instance: inband-vrf
  Prefix Length      Count
            /32          3
           /128          3
            Sum          6
Instance: ip2vrf
  Prefix Length      Count
            /24          1
            /32          2
           /128          2
            Sum          5
Instance: li-vrf
  Prefix Length      Count
            /32          2
           /128          2
            Sum          4
Instance: mgmt-vrf
  Prefix Length      Count
            /32          3
           /128          3
            Sum          6
Instance: radius-vrf
  Prefix Length      Count
            /24          1
            /32          2
           /128          2
            Sum          5
```

Show Route Hierarchy

The show route hierarchy command provides the complete information of a prefix at a given afi, safi, instance from its source, RIBD, and FIBD.

Syntax:

show route hierarchy <option>

Option	Description
instance <instance_name>	Name fof the instance.
<afi>	To show only the routes for a specific network protocol, such as IPv4 or IPv6.
<safi>	To view specific types of routes, such as IPv4 unicast or IPv6 multicast,received from neighbors.
<afi> <safi> detail	Shows information in detail for a specified afi or safi.
<afi> <safi> instance <instance-name>	Filter route entries for specified instance for a specifc afi and safi.
label	Filters route entries by a specific MPLS label.
prefix	Filters route entries by a specified IP prefix.

Example:

```
supervisor@rtbrick.net: cfg> show route hierarchy ipv4 unicast instance default
prefix 192.1.0.1/32
direct:
  prefix: 192.1.0.1/32
    nexthop set hash: 11de2090964a7b7176fc8add0c2c8b9cab7d22e52ff8f1a3
      nexthop hash: 16805dc42973c9ddb6289aac36994bbcb11e09ef6fbdc65b
        nexthop4: 192.1.0.1, exit interface: lo-0/0/0/10
isis:
  prefix: 192.1.0.1/32
  preference: 15
    nexthop set hash: 144ebc3d7e41a1cbf034f0c2c278a70df83d4717b1dfec6a
      nexthop hash: 45611a55482ca710251d15dd7f0bb89aba8819025e68029f
        exit interface: local
ribd:
  prefix: 192.1.0.1/32
  source: direct
    adjacency set hash: 11de2090964a7b7176fc8add0c2c8b9cab7d22e52ff8f1a3
      nexthop hash: 16805dc42973c9ddb6289aac36994bbcb11e09ef6fbdc65b
        nexthop4: 192.1.0.1, exit interface: lo-0/0/0/10
fibd:
  prefix: 192.1.0.1/32
  source: direct
  route status: |VPP Route ADD|-|-|VPP Route Mapping Failure|-|VPP Route ADJ
Failure|-|-|-|-|-|-|-|-|BCM Route ADD|-|-|-|-|-|-|-
  bcm status: Download success
  vpp status: None
    adjacency set hash: 11de2090964a7b7176fc8add0c2c8b9cab7d22e52ff8f1a3
      nexthop hash: 16805dc42973c9ddb6289aac36994bbcb11e09ef6fbdc65b
        exit interface: lo-0/0/0/10
```

Show Route Import

Syntax:

show route import <option>

Option	Description
import	The command shows all the routes imported from another instance.
detail	Shows detailed information for imported routes.
instance	Shows routing table information for a specific instance for imported routes.
ipv4	Shows route information for the IPv4 routing table for imported routes.
ipv6	Shows route information for the IPv4 routing table for imported routes.
source	Shows source of the routing information for the specified source.
summary	Shows the number of imported routes selected by RIB for each source.

Show Route L2VPN

Syntax:

show route l2vpn <option>

Option	Description
l2vpn	Without any option, the commands display all the L2VPN routes.
detail	Shows detailed information for l2vpn routes.
evpn-vpws	Shows routing table information for the EVPN-VPWS address family.
instance	Shows L2VPN routing table information for a specific instance

Option	Description
label <label>	Shows L2VPN routing table information for destination label.
prefix	Shows L2VPN routing table information for the destination prefix.
source	Shows source information for the L2VPN routes.
summary	Shows the number of L2VPN routes selected by RIBD for each source.
vpls-vpws	Shows the L2VPN routes from VPLS-VPWS address specified.

Show Route Resolution

The show route-resolution command displays the routes that were requested to be resolved for their nexthops. Otherwise, it shows the route is unresolved.

Syntax:

show route-resolution <options>

Option	Description
-	Without any option, the command shows the information for all requests and response tables side by side.
destination-instance	Shows the information for all requests and responses for a destination instance.
look-up instance	Shows lookup instance routes.
prefix	Shows routes for prefix 4 or prefix 6.
resolved	Shows resolved routes.
source	Shows source of requested source.
unresolved	Shows unresolved routes.

Example 1: Below is an example of the route resolution requested by the protocol BGP.

```
supervisor@rtbrick>LEAF01: op> show route-resolution
```

```
192:1::2, Source: bgp
  Destination instance: default, AFI: ipv4, SAFI: vpn-multicast
  Lookup      instance: default, AFI: ipv6, SAFI: labeled-unicast
  Covering Prefix: 192:1::2/128
    Interface        MAC Address         Nexthop
    ifl-0/1/70/13    e8:c5:7a:8f:56:63   fe80::eac5:7aff:fe8f:5663
fe80::eac5:7aff:fe8f:5663, Source: bgp
  Destination instance: default, AFI: ipv6, SAFI: unicast
  Lookup      instance: default, AFI: ipv6, SAFI: unicast
  Covering Prefix: fe80::eac5:7aff:fe8f:5663/128
    Interface        MAC Address         Nexthop
    ifl-0/1/70/13    e8:c5:7a:8f:56:63   fe80::eac5:7aff:fe8f:5663
fe80::eac5:7aff:fe8f:5663, Source: bgp
  Destination instance: default, AFI: ipv4, SAFI: vpn-unicast
  Lookup      instance: default, AFI: ipv6, SAFI: labeled-unicast
  Covering Prefix: fe80::eac5:7aff:fe8f:5663/128
    Interface        MAC Address         Nexthop
    ifl-0/1/70/13    e8:c5:7a:8f:56:63   fe80::eac5:7aff:fe8f:5663
fe80::eac5:7aff:fe8f:5663, Source: bgp
  Destination instance: default, AFI: ipv6, SAFI: labeled-unicast
  Lookup      instance: default, AFI: ipv6, SAFI: labeled-unicast
  Covering Prefix: fe80::eac5:7aff:fe8f:5663/128
    Interface        MAC Address         Nexthop
    ifl-0/1/70/13    e8:c5:7a:8f:56:63   fe80::eac5:7aff:fe8f:5663
fe80::eac5:7aff:fe8f:5663, Source: bgp
  Destination instance: default, AFI: ipv6, SAFI: vpn-unicast
  Lookup      instance: default, AFI: ipv6, SAFI: labeled-unicast
  Covering Prefix: fe80::eac5:7aff:fe8f:5663/128
    Interface        MAC Address         Nexthop
    ifl-0/1/70/13    e8:c5:7a:8f:56:63   fe80::eac5:7aff:fe8f:5663
```

Show Route rib-in

The show route rib-in command shows all advertised routes' information received and stored in the inbound RIB (Routing Information Base). These commands allow you to view routes based on specific criteria such as instance, source, address family (AFI), subsequent address family identifier (SAFI) and so on.

Syntax:

show route rib-in <options>

Option	Description
-	Without any option, the command shows the information for all routes in the routing information base.
detail	Provides a detailed view of routes matching the specified prefix in the inbound RIB.

Option	Description
<afi>	To show only the routes for a specific network protocol, such as IPv4 or IPv6.
<safi>	To view specific types of routes, such as IPv4 unicast or IPv6 multicast, received from neighbors.
instance <instance>	Shows the routes in the inbound RIB of a specific routing instance.
source <source>	Shows the routes in the inbound RIB from a specific neighbor.
summary	Provides a summary view of all routes in the inbound RIB.

Show Route Prefix

The show route prefix command shows all advertised routes' information received and stored in the inbound RIB (Routing Information Base).

Syntax:

show route prefix <options>

Option	Description
-	Without any option, the command shows the information for all routes in the routing information base.
prefix <value>	Specify the prefix. This is to see if a specific route with a particular prefix has been received from a neighbor.
rib-in	Shows all routes currently stored in the routing information base without any filtering.
detail	Provides a detailed view of routes matching the specified prefix in the inbound RIB.

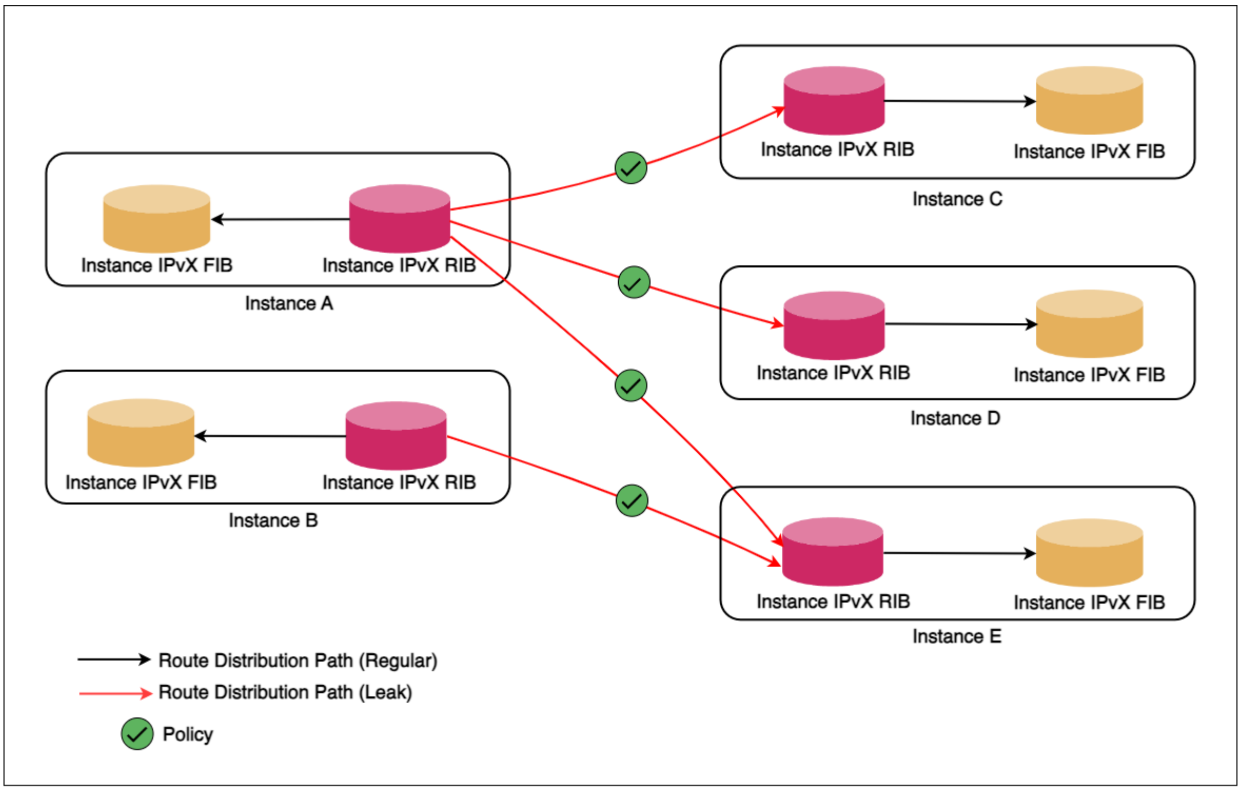
2.2. Route Leaking

2.2.1. Route Leaking Overview

Route leaking involves exporting a route from one routing instance to another routing instance.

As shown in the figure below, RBFS allows importing routes from the RIB of one instance to another instance. Additionally, the routes can be selectively imported based on the associated policy.

Route Leaking Across Instances



Guidelines and Limitations

- Currently, the route leaking “import-rib” feature is limited to importing only IPv4 and IPv6 unicast routes.

Supported Platforms

The Route Leaking feature is supported by the Multiservice Edge images for the following Q2A and Q2C hardware platforms.

Q2A:

- Edgecore CSR440 (AS7535-28XB)

- UfiSpace S9510-28DC

Q2C:

- Edgecore AGR420 (AS7946-74XKSB)
- UfiSpace S9600-72XC
- UfiSpace S9600-102XC

Route Leaking Policy Attributes and Match Types

The following table provides all the supported route leaking policy attributes and match types.

Attribute Type	Match Types Supported
ipv4-prefix	regex exact longer or-longer prefix-length-exact prefix-length-greater prefix-length-greater-or-exact
ipv6-prefix	regex exact longer or-longer prefix-length-exact prefix-length-greater prefix-length-greater-or-exact
community	regex exact exists
extended-community	regex exact exists
large-community	regex exact exists

Attribute Type	Match Types Supported
source	regex exact

2.2.2. Route Leaking Configuration

Configuring Route Leaking from one instance to another

To configure route leaking from one instance to another, configure the "import-rib" for the routing instance (AFI/SAFI).

Syntax:

set instance <destination-instance> **address-family** <afi> <safi> **import-rib** <source-instance>

Attribute	Value
<destination-instance>	Name of the routing instance to which routes will be imported to.
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6.
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast.
<source-instance>	Specifies the routing instance from which the routes will be imported.

In the following example, IPv4 and IPv6 unicast routes will be copied from the “vrf-blue” to “vrf-red” instance RIB.

```
set instance vrf-red address-family ipv4 unicast import-rib vrf-blue
set instance vrf-red address-family ipv6 unicast import-rib vrf-blue
```

The following is an example of JSON output showing the instance configuration.

```
supervisor@rtbrick>LEAF01: cfg> show config instance vrf-red
{
  "rtbrick-config:instance": [
    {
      "name": "vrf-red",
      "address-family": [
        {
          "afi": "ipv4",
          "safi": "unicast",
          "import-rib": [
            {
              "instance": "vrf-blue"
            }
          ]
        },
        {
          "afi": "ipv6",
          "safi": "unicast",
          "import-rib": [
            {
              "instance": "vrf-blue"
            }
          ]
        }
      ]
    }
  ]
}
```

Configuring Route Leaking with Policy

You can attach a policy to import selective routes as per the required criteria.

Syntax:

set instance <destination-instance> **address-family** <afi> <safi> **import-rib** <source-instance> **policy** <policy-name>

Attribute	Value
<destination-instance>	Name of the routing instance to which routes will be imported to.
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6.
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast.
<source-instance>	Name of the routing instance to which routes will be imported.
<policy-name>	Name of the routing instance to which routes will be imported based on the specified policy.

In the following example, the policy match_ipv4_prefix is attached to the 'vrf-blue' instance.

```
set instance vrf-red address-family ipv4 unicast import-rib vrf-blue policy
match_ipv4_prefix
```

Example: The following is an example of JSON output of the above configuration.

```
{
  "rtbrick-config:instance": [
    {
      "name": "vrf-red",
      "address-family": [
        {
          "afi": "ipv4",
          "safi": "unicast",
          "import-rib": [
            {
              "instance": "vrf-blue",
              "policy": "match_ipv4_prefix"
            }
          ]
        }
      ]
    }
  ]
}
```

2.2.3. Route Leaking Operational Commands

show route import

The show route import command displays all the imported routes.

Syntax:

show route import <option>

Attribute	Value
detail	Displays detailed route information of the imported routes.
instance <instance-name>	Displays route information for the imported routes for the specified instance.
instance <instance-name> detail	Displays detailed route information for the imported routes for the specified instance.
instances <instance-name> prefix <prefix>	Displays route information for the imported routes for the specified instance and prefix.

Attribute	Value
instance <instance-name> prefix <prefix> detail	Displays detailed route information for the imported routes for the specified instance and prefix.
instance <instance-name> summary	Displays summarized route information for the imported routes for the specified instance.
ipv4	Displays route information for the IPv4 address family.
ipv6	Displays route information for the IPv6 address family.
source	Displays route information for the specified route source.
summary	Display summarized route information for the imported routes.

Example 1: Source instance (vrf-blue) RIB

```
supervisor@rtbrick>LEAF01: op> show route ipv4 unicast source bgp instance vrf-
blue
Flags: & - Imported
Instance: vrf-blue, AFI: ipv4, SAFI: unicast
Flags  Prefix/Label        Source        Pref     Next Hop     Outgoing Interface
       192.168.0.20/32     bgp           20       12.0.0.2     ifl-0/0/2/0
       192.168.0.21/32     bgp           20       12.0.0.2     ifl-0/0/2/0
       192.168.1.20/32     bgp           20       12.0.0.2     ifl-0/0/2/0
       192.168.1.21/32     bgp           20       12.0.0.2     ifl-0/0/2/0
       192.168.4.20/32     bgp           20       12.0.0.2     ifl-0/0/2/0
supervisor@rtbrick>LEAF01: op>
```

Example 2: Destination instance RIB

```
supervisor@rtbrick>LEAF01: op> show route import ipv4 unicast source bgp instance
vrf-red
Flags: & - Imported
Instance: vrf-red, AFI: ipv4, SAFI: unicast
Flags  Prefix/Label      Source   Pref     Next Hop     Outgoing Interface
&      192.168.0.20/32   bgp      20       12.0.0.2     ifl-0/0/2/0
&      192.168.0.21/32   bgp      20       12.0.0.2     ifl-0/0/2/0
&      192.168.1.20/32   bgp      20       12.0.0.2     ifl-0/0/2/0
&      192.168.1.21/32   bgp      20       12.0.0.2     ifl-0/0/2/0
&      192.168.4.20/32   bgp      20       12.0.0.2     ifl-0/0/2/0
supervisor@rtbrick>LEAF01: op>
```

Example 3: Detailed view of the destination instance (vrf-red) RIB

```
supervisor@rtbrick>LEAF01: op> show route import ipv4 unicast source bgp instance
vrf-red detail
Instance: vrf-red, AFI: ipv4, SAFI: unicast
```

```
192.168.0.20/32
  Source: bgp, Preference: 20
  Import type: inter-instance, Import instance: vrf-blue
    Next Hop: 12.0.0.2
      Covering prefix: 12.0.0.2/32
      Next Hop type: direct, Next Hop action: None
      Resolved in: default-ipv4-labeled-unicast
      Outgoing Interface: ifl-0/0/2/0, NextHop MAC: 7a:02:bb:c0:00:00
192.168.0.21/32
  Source: bgp, Preference: 20
  Import type: inter-instance, Import instance: vrf-blue
    Next Hop: 12.0.0.2
      Covering prefix: 12.0.0.2/32
      Next Hop type: direct, Next Hop action: None
      Resolved in: default-ipv4-labeled-unicast
      Outgoing Interface: ifl-0/0/2/0, NextHop MAC: 7a:02:bb:c0:00:00
192.168.1.20/32
  Source: bgp, Preference: 20
  Import type: inter-instance, Import instance: vrf-blue
    Next Hop: 12.0.0.2
      Covering prefix: 12.0.0.2/32
      Next Hop type: direct, Next Hop action: None
      Resolved in: default-ipv4-labeled-unicast
      Outgoing Interface: ifl-0/0/2/0, NextHop MAC: 7a:02:bb:c0:00:00
192.168.1.21/32
  Source: bgp, Preference: 20
  Import type: inter-instance, Import instance: vrf-blue
    Next Hop: 12.0.0.2
      Covering prefix: 12.0.0.2/32
      Next Hop type: direct, Next Hop action: None
      Resolved in: default-ipv4-labeled-unicast
      Outgoing Interface: ifl-0/0/2/0, NextHop MAC: 7a:02:bb:c0:00:00
192.168.4.20/32
  Source: bgp, Preference: 20
  Import type: inter-instance, Import instance: vrf-blue
    Next Hop: 12.0.0.2
      Covering prefix: 12.0.0.2/32
      Next Hop type: direct, Next Hop action: None
      Resolved in: default-ipv4-labeled-unicast
      Outgoing Interface: ifl-0/0/2/0, NextHop MAC: 7a:02:bb:c0:00:00
supervisor@rtbrick>LEAF01: op>
```

Example 4: Destination instance (vrf-red) RIB after applying the policy (match_ipv4_prefix)

```
supervisor@rtbrick>LEAF01: op> show route import ipv4 unicast source bgp instance
vrf-red
Flags: & - Imported
Instance: vrf-red, AFI: ipv4, SAFI: unicast
Flags  Prefix/Label       Source     Pref    Next Hop  Outgoing Interface
&      192.168.1.20/32    bgp        20      12.0.0.2  ifl-0/0/2/0
```

In the above example, the below policy is used to import the selective route.

```
set policy statement match_ipv4_prefix
set policy statement match_ipv4_prefix ordinal 1
set policy statement match_ipv4_prefix ordinal 1 match
set policy statement match_ipv4_prefix ordinal 1 match rule 1
set policy statement match_ipv4_prefix ordinal 1 match rule 1 type ipv4-prefix
set policy statement match_ipv4_prefix ordinal 1 match rule 1 value-type discrete
set policy statement match_ipv4_prefix ordinal 1 match rule 1 match-type exact
set policy statement match_ipv4_prefix ordinal 1 match rule 1 value
192.168.1.20/32
set policy statement match_ipv4_prefix ordinal 1 action
set policy statement match_ipv4_prefix ordinal 1 action rule 1
set policy statement match_ipv4_prefix ordinal 1 action rule 1 operation return-
permit
```

The following is an example of JSON output showing the above policy configuration.

```
{
  "rtbrick-config:statement": [
    {
      "name": "match_ipv4_prefix",
      "ordinal": [
        {
          "ordinal": 1,
          "match": {
            "rule": [
              {
                "rule": 1,
                "type": "ipv4-prefix",
                "value-type": "discrete",
                "match-type": "exact",
                "value": [
                  "192.168.1.20/32"
                ]
              }
            ]
          },
          "action": {
            "rule": [
              {
                "rule": 1,
                "operation": "return-permit"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

2.3. BGP

2.3.1. BGP Overview

BGP is a standard exterior gateway protocol (EGP) supported by RtBrick. BGP is considered a “Path Vector” routing protocol and maintains a separate routing table based on the shortest Autonomous System (AS) path and various other route attributes.

Supported BGP Standards

RFC Number	Description
RFC 2545	Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing
RFC 2918	Route Refresh Capability for BGP-4
RFC 4271	A Border Gateway Protocol 4 (BGP-4)
RFC 4364	BGP/MPLS IP Virtual Private Networks (VPNs)
RFC 4456	BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)
RFC 4486	Subcodes for BGP Cease Notification Message
RFC 4760	Multiprotocol Extensions for BGP-4
RFC 5492	Capabilities Advertisement with BGP-4
RFC 6793	BGP Support for Four-Octet Autonomous System (AS) Number Space
RFC 6608	Subcodes for BGP Finite State Machine Error
RFC 6774	Distribution of Diverse BGP Paths [Partial Support]

RFC and draft compliance are partial except as specified.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Supported BGP Features

The RBFS supports the following BGP functions:

- Basic BGP Protocol
- Multiprotocol extension for BGP
- Multipath for iBGP and eBGP
- Four-byte AS numbers
- Nexthop Self or next-hop unchanged
- Fast external-failover
- Route reflection
- MD5 Authentication
- Generalized TTL Security Mechanism (GTSM)
- Route Refresh
- Advanced route refresh
- Route redistribution
- Multihop EBGP
- Route selection flexibility (always compare MED, ignore AS Path, and so on)
- Add path
- Hostname/Domain name
- Dynamic peers
- Community, Extended Community, and Large Community support
- 6PE Support
- Curated Upstream Path Selection

The statements and commands required to configure and verify the functioning of BGP features are described in this guide.

MD5 Authentication

BGP supports the authentication mechanism using the Message Digest 5 (MD5) algorithm. When authentication is enabled, any Transmission Control Protocol (TCP) segment belonging to BGP exchanged between the peers is verified and accepted only if authentication is successful. For authentication to be successful, both peers must be configured with the same password. If authentication fails, the BGP neighbor relationship is not established.

GTSM

The Generalized TTL Security Mechanism (GTSM) protects a BGP session by evaluating the Time to Live (TTL) value contained in the IP header of incoming BGP packets.

TTL security is enabled using the ttl-security command on a single-hop BGP session. When TTL security is enabled on a BGP session the IP TTL values in packets is set to 255. To enable TTL security on a multihop BGP session, enable ttl-security and configure ttl-limit to match the expected TTL value.

For details about configuring GTSM, see the Peer Group Configuration section.

IPv6 Provider Edge (6PE)

The Provider Edge (6PE) solution enables IPv6 communication over the MPLS IPv4 core network. IPv6 reachability information is associated with a label and transferred through MP-BGP(AFI: 2 SAFI:4). IPv4 mapped IPv6 address is used to encode the next-hop information. The edge nodes in the MPLS IPv4 core have to support both IPv4 and IPv6. The IPv6 Labeled Unicast routes received from the 6PE peer is considered as IPv6 unicast routes and installed in IPv6 Unicast FIB. The received Label is attached to the IPv6 data traffic at the Ingress node and tunneled through an MPLS tunnel(SR) to the egress node, the label identifies the IPv6 traffic, and the egress node would POP the label and forward the ipv6 traffic towards the destination.

Policies

The Role of a Routing Policy

Routing Policies are the rules that allow you to control and modify the default behavior of the routing protocols such as BGP and IS-IS. To use routing policies, you configure policies and then apply policies to peer groups or instances.

Attachment Points

Policies are useful when they are applied to routes, for which they need to be made known to routing protocols. In BGP, for example, there are several situations where policies can be used, the most common of these is defining import and export policy. The policy attachment point is the point in which an association is formed between a specific protocol entity, in this case, a BGP neighbor, and a specific named policy.

RtBrick supports attaching a BGP routing policy at two levels:

- Peer group address-family level
- Instance address-family level

In each case, you can apply the policy as an import or export policy and filter. As expected, import filters determine which routing updates are accepted and export filters determine which routes are advertised to other peers.

Policy Processing

An import policy, when applied to an address family at the peer group level, examines all *incoming* routes from all BGP peers in the peer group, but only for that address family.

An export policy, when applied to an address family at the peer group level, examines all outgoing routes to all BGP peers in the peer group, but only for that address family.

At the instance level, routing policies that are applied to an address family can work as import or export policies, but for the instances as a whole.

An import policy, when applied to an address family at the instance level, examines all incoming routes before accepting the information only from global or default tables to other instances or VRF tables.

An export policy, when applied to an address family at the instance level, examines all outgoing routes before sending the information from the VRF to global, and then to the VPN table (default).

BGP Communities

BGP communities are a way to tag routes with additional information that can be used for routing decisions and routing policies. A BGP community is a collection of destinations that share a common property. It allows you to apply actions to that group of destinations as a whole, rather than to each member individually. BGP communities are used to provide an additional capability to tag routes and thereby, modify BGP routing policies.

Communities can influence routing decisions such as accept, reject, preference, and redistribution. RBFS allows various operations such as appending, modifying, or deleting communities as the route travels from one router to another. A BGP community is an optional attribute that can be tagged with a route. The AS path attribute contains community information, which is used to identify community members. It allows for route manipulation without requiring multiple route filters.

RBFS supports three types of BGP communities for implementing complex routing policies and effectively controlling traffic across diverse and large-scale networks.

Standard Communities

These are used to control routing policies and are represented by a 32-bit number. They include well-known Communities such as no-export, no-advertise, and so on.

Extended Communities

Extended Communities provide functionalities beyond what standard communities offer. They are represented by 64-bit number and are displayed as two 32-bit numbers. They include additional fields such as type and sub-type, providing more granularity and the ability to carry additional information. Types: These include route targets, route origin, and other specialized community types that extend.

Large Communities

Represented as a 96-bit number, large communities consist of three 32-bit numbers separated by colons (for example, 64512:100:200). They provide space for tagging routes with extensive information and are easier to use in global policies. Large communities are used in large-scale deployments due to the larger addressing space.

Support for Allow As In

The Allow AS In feature enables BGP routers to receive and process routes even if the routes carry the router’s autonomous system number in the 'AS-Path' attribute. Usually, a BGP router discards routes if the ‘AS-Path’ attribute contains its autonomous system number to prevent loops. However, with the "Allow AS In" feature, such routes can be accepted and routed to the destination without being rejected. You can specify the allow-as-in option value in the peer configuration. For example, if you specify the value ‘5’, the router accepts BGP routes that contain its own AS number (in the AS path) up to 5 times.

BGP Best Path Selection Algorithm

BGP routers typically receive multiple paths to the same destination. A BGP router forms a neighbor relationship by connecting to its neighbors and exchanging the routes, once the connection is established. The BGP route selection algorithm decides which is the best path to install in the IP routing table and to use for traffic forwarding.

BGP Best Path Selection Algorithm

The algorithm eliminates all routes whose next hop is not reachable. Circular route resolution is considered for route resolution.

The algorithm for determining all the routes that have the same route prefix is as follows:

1. The first route selection is performed based on the lowest route source. Route from the local route source is always preferred over the received route. For example, when there is the same prefix route that is redistributed and received from a neighbor, the local (redistributed route) is always preferred. The locally learned route is preferred over the locally crossed or remote crossed route (in the case of VPN, a route might be learned locally in the VRF. The same prefix might be received from the remote as VPNv4. After importing into the VRF routing table, a locally learned route is preferred over the remote local crossed route).

2. Prefer the path with the highest local preference if the route source is the same. If a path does not have a local preference attribute (for example, it is received from an eBGP peer), then it is considered to have the local preference assigned in the given BGP instance. The show bgp summary command shows the local preference assigned in the system. This can be changed using the set local-preference value.

3. Prefer the route with the shortest AS path, if no route originated. If there is no AS_PATH attribute, then it is assumed to be of length 0. A single AS_SET is considered to be a length of 1.

4. Prefer the path with the lowest origin type, if the AS path length is the same as all the paths. The available three values include IGP, EGP and Incomplete. The lowest value is IGP and the highest value is Incomplete.

5. Prefer the path with the lowest Multi Exit Discriminator (MED), if the original codes are the same. (By default, MED values are only compared when routes are learned from the same AS. Routes without MED values are treated as if they have a MED value of 0, which is the lowest and, therefore, always the most preferred value.)
6. Prefer external BGP learned routes over internal BGP routes at this point after comparing the route type (internal BGP and external BGP).
7. Prefer the path whose next hop is resolved through the IGP route with the lowest metric.
8. Prefer the length with a shorter CLUSTER length path. If the CLUSTER attribute is not present, the length is assumed to be 0.
9. Prefer the path from the peer with the lowest router ID. For any path with an originator ID attribute, substitute the originator ID for the router ID during router ID comparison.
10. Prefer the lowest peer IP address as the tie-breaker, if the router-id is the same for both sessions. This is for BGP to make route selections in case of multiple peerings are used between the same routers.
11. If add path is enabled, then the same peer might advertise multiple paths for the same prefix. The path with a lower send path ID is preferred.

The BGP best path selection algorithm also provides a mechanism to discard paths that are not considered candidates for the best path. The following paths are discarded:

- The paths for which next-hops are not resolved.
- The paths originated from an eBGP neighbor if a local AS is shown in the AS-PATH attribute.
- If the BGP enforce-first-as attribute is enabled and the update does not contain the AS number of the neighbor as the first AS number in the AS-SEQUENCE attribute.
- The paths which are marked as Received-only.

Support for Upstream BGP Route Export into VRFs of Peering Customers

This feature enables service providers to export or leak the upstream BGP routes into the VRFs for advertisement to peering customers. Certain peering customers usually do not require all the upstream BGP routes that the service provider receives. Instead, they want routes only from specific upstream providers. This feature gives peering customers more control over the upstream transit providers and they can choose their preferred upstream neighbor.

It allows service providers to selectively leak selected BGP routes from one instance to another without having to install the routes in the forwarding path in order to conserve the forwarding hardware resources. The leaked or exported routes do not get installed in peering customer VRFs. The feature offers an option to install the default route in the customer VRF that points to the upstream neighbor. It enables forwarding customer traffic towards the internet, connecting the customer site with external networks without requiring each specific route to be installed.

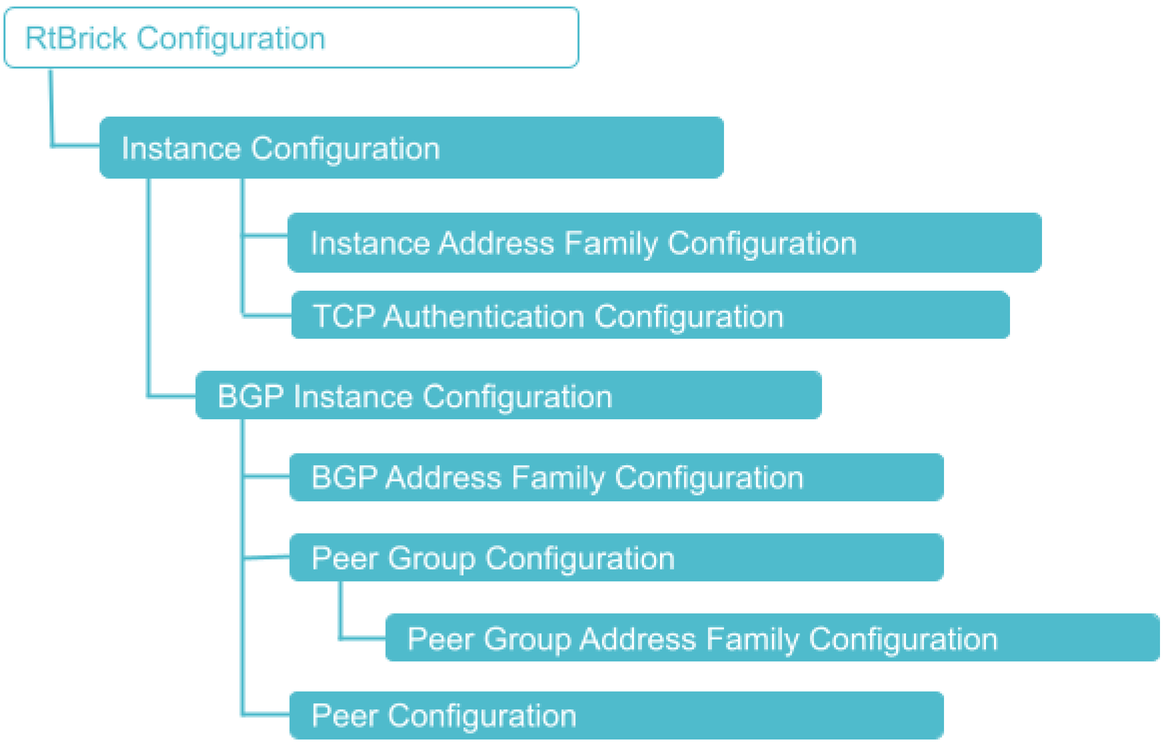
Policy to Notify BGP of Nexthop Changes for VPN Route Label Updates

The nexthop change via policy is notified to BGP and this will trigger a change in label. This notification enables BGP to update the labels for those routes.

2.3.2. BGP Configuration

Configuration Hierarchy

The diagram illustrates the BGP configuration hierarchy. All BGP configuration is done within an instance, for example the default instance or a VPN service instance. The instance configuration hierarchy includes parameters required for BGP but not part of the BGP configuration hierarchy itself. The BGP instance configuration hierarchy includes parameters which are generic to the respective BGP instance. The sub-hierarchies include parameters which are specific to address families, peer groups, and peers.


RtBrick Configuration
Instance Configuration
Instance Address Family Configuration
TCP Authentication Configuration
BGP Instance Configuration
BGP Address Family Configuration
Peer Group Configuration
Peer Group Address Family Configuration
Peer Configuration

Configuration Syntax and Commands

The following sections describe the BGP configuration syntax and commands.

Instance Configuration

The instance configuration hierarchy includes parameters that are required for or used by BGP, but that are not part of the BGP protocol configuration hierarchy itself.

Route distinguishers and router IDs are configured directly at the instance hierarchy.

Syntax:

set instance <instance-name> <attribute> <value>

Attribute	Description
route-distinguisher <as-number|ipv4-address:id>	The route distinguisher (RD) uniquely defines routes within an IPv4 network. PE routers use route distinguishers to identify which VPN a packet belongs to. Supported formats are <as-number:id> or <ipv4-address:id>. ℹ If you want to use the format <as-number:id> with a 4-byte ASN, specify it with an "L". For example, set instance services route-distinguisher 4200000000L:101
ipv4-router-id <ipv4-address>	The router ID of the routing instance.

Example: Instance Identifier Configuration

```
supervisor@leaf1: cfg> show config instance services
{
  "rtbrick-config:instance": {
    "name": "services",
    "ipv4-router-id": "198.51.100.41",
    "route-distinguisher": "198.51.100.41:101",
    <...>
  }
}
```

Address Families

At the instance address family hierarchy, you can enable or disable address families for the instance, and configure parameters like route targets.

Please note default settings depend on the instance. For the 'default' instance, the IPv4 and IPv6 unicast, multicast, and labeled unicast, as well as the MPLS unicast address families are enabled by default. For any non-default instance, no address family is enabled by default and needs to be enabled by configuration.

Syntax:

set instance <instance-name> **address-family** <afi> <safi> <attribute>

```
  <value>
```

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6, or mpls
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast, labeled-unicast, or multicast
route-target (import | export) <rt-value>	Route targets (RT) are used to transfer routes between VPN instances. The RT identifies a subset of routes that should be imported to or exported from a particular VPN instance. You can configure a RT for importing or exporting routes or both. If you want to use the format <as-number:id> with a 4-byte ASN, specify it with an "L". For example, set instance services address-family ipv4 unicast route-target export target:4200000000L:14
policy (import | export) <policy-name>	There are two attachment points for BGP policies. At this configuration hierarchy, you can attach import or export policies to the instance. These policies apply when routes are imported from the BGP protocol into the instance, or exported from the instance to the BGP protocol.

Example: Instance Address Family Configuration

```
supervisor@leaf1: cfg> show config instance services
{
  "rtbrick-config:instance": {
    "name": "services",
    <...>
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "unicast",
        "policy": {
          "export": "MY_V4_POLICY"
        },
        "route-target": {
          "import": "target:198.51.100.70:14",
          "export": "target:198.51.100.70:14"
        }
      },
      {
        "afi": "ipv6",
        "safi": "unicast",
        "policy": {
          "export": "MY_V6_POLICY"
        },
        "route-target": {
          "import": "target:198.51.100.70:16",
          "export": "target:198.51.100.70:16"
        }
      }
    ],
    <...>
  }
}
```

TCP Authentication Configuration

In the instance TCP authentication hierarchy, you can optionally enable MD5 or HMAC-SHA-1-96, HMAC-SHA-256-128, or AES-128-CMAC-96 authentication. Authentication is not configured for BGP directly but for the TCP sessions used by BGP. It is necessary to bind authentication to a peer in order for the authentication to work. For details about configuring a BGP Peer, see the section Peer Configuration.

BGP TCP authentication is not backward compatible.

Syntax:

set instance <instance> **tcp authentication** <authentication-id> <attribute> <value>

Attribute	Description
<authentication-id>	Authentication identifier
type <type>	Authentication identifiers such as MD5
type <type>	Authentication identifiers such as MD5 or HMAC-SHA-1-96,HMAC-SHA-256-128 or AES-128-CMAC-96
key1-id <key1-id>	Key ID1 of the receiver
key1-encrypted-text <key1-encrypted-text>	Encrypted text of key1

Attribute	Description
key1-plain-text <key1-plain-text>	Plain text of key1
key2-id <key2-id>	Key ID2 of the receiver
key2-encrypted-text <key2-encrypted-text>	Encrypted text of key2
key2-plain-text <key2-plain-text>	Plain text of key2

Example: BGP TCP Authentication Configuration

```
{
    "rtbrick-config:tcp": {
      "authentication": [
        {
          "authentication-id": "auth1",
          "type": "MD5",
          "key1-id": 10,
          "key1-encrypted-text": "$2784cfa7523916c8cc5dfeba83562cbb4",
          "key2-id": 20,
          "key2-encrypted-text": "$2e9bb845e3cfcf8173973029e5c1d90d6"
        }
      ]
    }
  }
```

BGP Instance Configuration

At this configuration hierarchy, you configure BGP protocol parameters which are generic to the BGP instance.

Syntax:

set instance <instance-name> **protocol bgp** <attribute> <value>

Attribute	Description
host-name <host-name>	The name of the BGP host, to a maximum of 64 characters
domain-name <domain-name>	The name of the BGP routing domain, to a maximum of 64 characters

Attribute	Description
enforce-first-as <enable|disable>	By default, the BGP routing process enforces the First AS feature. It discards updates received from an eBGP peer if the peer does not list its own AS number as the first segment in the AS_PATH BGP attribute. Disable the First AS feature to accept updates without the peer’s source AS matching the first AS in the AS_PATH attribute.
local-as <as-number>	The AS number in four-byte format. The numbers allowed are from 1 to 4294967295.
local-preference <preference-value>	The local preference for the BGP protocol. The numbers allowed are from 0 to 4294967295. The local preference is used to select the exit path for an AS.
med <med-value>	The BGP Multi-Exit Discriminator (MED) value. The numbers allowed are from 0 to 4294967295. When an AS has multiple links to another AS, the MED value is used to determine the exit to use to reach the other AS.
protocol-preference (internal | external) <preference-value>	Protocol preference of routes learned by eBGP ('external'), iBGP ('internal'), or both. This preference is used to select routes learned from multiple protocols.
router-id <router-id>	Router identifier in IPv4 format
cluster-id <cluster-identifier>	The cluster ID associates routers in a group within a BGP routing instance. Routers belong to the same cluster if they have the same cluster-ID. The cluster ID is formatted as an IPv4 address.
timer hold-time <seconds>	Hold timer in seconds.
timer keepalive <seconds>	Keep a live timer in seconds.

Example: BGP Instance Configuration

The following example shows some global BGP instance configuration attributes. The further BGP configuration like peer groups and peers is shown in the examples in the subsequent sections.

```
supervisor@spine1: cfg> show config instance default protocol bgp
{
  "rtbrick-config:bgp": {
    "cluster-id": "198.51.100.51",
    "domain-name": "rtbrick.com",
    "host-name": "spine1",
    "local-as": 4200000100,
    "local-preference": 50,
    "router-id": "198.51.100.51",
    "protocol-preference": {
      "internal": 180,
      "external": 20
    },
    "timer": {
      "hold-time": 30,
      "keepalive": 10
    },
    <...>
}
```

BGP Address Family Configuration

This configuration hierarchy refers to parameters that are specific to address families but generic to the BGP instance, as opposed to peer-group specific address families configuration. At this hierarchy, you can enable or disable address families for BGP, and configure various features specific to the address family.

RBFS offers support for a configurable route resolution policy. Initially, RBFS will attempt to resolve the BGP routes using primary resolve-nexthop afi/safi configuration. In case of failure, RBFS will then proceed to resolve the BGP routes using the secondary-nexthop afi/safi configuration.

Syntax:

set instance <instance-name> **protocol bgp address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, or ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast, labeled-unicast, vpn-unicast, multicast, or vpn-multicast

Attribute	Description
default-information originate <true|false>	Generate and distribute a default route information
download-count <count>	Forward packets over multiple paths, set maximum prefixes to use
multipath <number>	Enable load sharing among multiple BGP paths
retain-route-target (enable|disable)	Retain VPN routes for all route targets, by default this feature is enabled
resolve-nexthop afi <afi>	Address family to resolve the next-hop
resolve-nexthop safi <safi>	Sub-address family to resolve the next-hop
resolve-nexthop-secondary afi <afi>	Address family to resolve the next-hop when the primary route resolution fails
resolve-nexthop-secondary safi <safi>	Sub-address family to resolve the next-hop when the primary route resolution fails
redistribute <source>	Enable the redistribution feature to dynamically inject specific types of routes into the BGP protocol. Supported route sources are direct, igmp, ipoe, isis, ospf, pim, ppp, static, and arp-nd.
redistribute <source> policy <policy>	Attach a policy to the redistribution process
srgb base <value>	Segment Routing Global Block (SRGB) start label. The SRGB is the range of label values reserved for segment routing (SR). These values are assigned as segment identifiers (SIDs) to SR-enabled network nodes and have global significance throughout the routing domain. SRGB is supported for labeled unicast only.
srgb index <value>	Segment Routing Global Block (SRGB) index
srgb range <value>	Segment Routing Global Block (SRGB) label range

Example 1: BGP Address Family Configuration with Segment Routing

```
supervisor@spine1: cfg> show config instance default protocol bgp
{
  "rtbrick-config:bgp": {
    <...>
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "vpn-unicast"
      },
      {
        "afi": "ipv6",
        "safi": "labeled-unicast",
        "srgb": {
          "base": 5000,
          "range": 1000,
          "index": 11
        }
      },
      {
        "afi": "ipv6",
        "safi": "unicast"
      },
      {
        "afi": "ipv6",
        "safi": "vpn-unicast"
      }
    ],
    <...>
  }
}
```

Example 2: BGP Address Family Configuration with Redistribution

```
supervisor@leaf1: cfg> show config instance services protocol bgp
{
  "rtbrick-config:bgp": {
    <...>
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "unicast",
        "redistribute": [
          {
            "source": "direct"
          },
          {
            "source": "ppp"
          },
          {
            "source": "arp-nd"
          },
          {
            "source": "static"
          }
        ]
      },
      {
        "afi": "ipv6",
        "safi": "unicast",
        "redistribute": [
          {
            "source": "direct"
          },
          {
            "source": "ppp"
          },
          {
            "source": "static"
          }
        ]
      }
    ]
  }
}
```

Example 3: BGP Address Family Configuration with Redistribution and Redistribution Policy

```
supervisor@leaf1: cfg> show config instance services protocol bgp
{
  "rtbrick-config:bgp": {
    <...>
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "unicast",
        "redistribute": [
          {
            "source": "direct"
            "policy": "MY_REDISTRIBUTION_POLICY"
          },
          {
            "source": "ppp"
          },
          {
            "source": "static"
          }
        ]
      },
      {
        "afi": "ipv6",
        "safi": "unicast",
        "redistribute": [
          {
            "source": "direct"
            "policy": "MY_REDISTRIBUTION_POLICY"
          },
          {
            "source": "ppp"
          },
          {
            "source": "static"
          }
        ]
      }
    ]
  }
}
```

Example 4: BGP Address Family with Resolve Nexthop Configuration.

In the example below, RBFS would resolve the BGP routes in the IPv4 labeled-unicast RIB. If this fails, then the routes would be resolved in IPv4 unicast RIB.

```
supervisor@leaf1: cfg> show config instance default protocol bgp address-family
ipv4 unicast
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "unicast",
      "resolve-nexthop": {
        "afi": "ipv4",
        "safi": "labeled-unicast"
      },
      "resolve-nexthop-secondary": {
        "afi": "ipv4",
        "safi": "unicast"
      }
    }
  ]
}
```

Peer Group Configuration

Peer Groups

In BGP, neighbor peers with the same update policies can be grouped to simplify the initial configuration and updates. Peers share the same policies such as route maps, distribution lists, filter lists, update sources, and so on, so peer groups only need one configuration statement for these values.

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> <attribute> <value>

Attribute	**Description**
local-as <as-number>	Local AS number for the peer group
remote-as <as-number>	Remote AS number for the peer group
any-as <true|false>	Enable dynamic AS negotiation for this peer group

Attribute	Description
ebgp-multihop <hop-count>	By default, the maximum number of hops between eBGP peers is 1 (direct connection). This hop count overrides the default behavior allowing connectivity between eBGP peers not directly connected.
link-local-nexthop-only <true|false>	Enable BGPv6 peerings using the IPv6 link-local addresses
no-prepend <true|false>	Do not prepend the local AS for advertisements to the peer
replace-as <true|false>	Prepend only the local AS for advertisements to the peer
ttl-security <enable|disable>	Enables or disalbes Generalized TTL Security Mechanism (GTSM).
ttl-limit <ttl-limit>	Specifies the minimum TTL value of packets from the BGP neighbor for TTL Security. The valid range is 1 to 255.

Address Families

At this configuration hierarchy, you can enable the address families that shall be supported for the group peers, and enable features specific to the address family. By default, BGP neighbor sessions support the IP4v unicast and multicast address families.

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, or ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast, labeled-unicast, vpn-unicast, multicast, or vpn-multicast

Attribute	Description
add-path	Negotiate additional path capabilities with these peers, so that more than one path can be active to the peers in the group
default-information originate <true|false>	Generate and advertise a default route to peers in the group
extended-nexthop	Enable extended-next-hop encoding for BGP peer groups to allow the transfer of IPv4 prefixes over an IPv6 connection
nexthop-self <true|false>	Set the advertised BGP nexthop to yourself, this is the default for eBGP
nexthop-unchanged <true|false>	Do not modify the advertised BGP nexthop, this is the default for iBGP
update-nexthop (ipv4-address | ipv6-address) <address>	BGP nexthop address for routes advertised to this peer group
remove-private-as <true|false>	Remove private AS numbers from routes advertised to group peers
route-reflect-client <true|false>	Configure this peer as a route reflector client
policy (import | export) <policy-name>	Apply a routing policy to the peer group

Example 1: BGP Peer Group Configuration

```
supervisor@leaf1: cfg> show config instance default protocol bgp peer-group spine
{
  "rtbrick-config:peer-group": {
    "pg-name": "spine",
    "link-local-nexthop-only": "true",
    "remote-as": 4200000100,
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "vpn-unicast",
        "extended-nexthop": "true",
        "update-nexthop": {
          "ipv6-address": "2001:db8:0:19::"
        }
      },
      {
        "afi": "ipv6",
        "safi": "labeled-unicast"
      },
      {
        "afi": "ipv6",
        "safi": "unicast"
      },
      {
        "afi": "ipv6",
        "safi": "vpn-unicast",
        "update-nexthop": {
          "ipv6-address": "2001:db8:0:19::"
        }
      }
    ]
  }
}
```

Example 2: GTSM Configuration

```
{
  "rtbrick-config:peer-group": [
    {
      "pg-name": "ipv4_bgp",
      "remote-as": 4200000100,
      "ttl-security": "enable",
      "ttl-limit": 253,
      "address-family": [
        {
          "afi": "ipv4",
          "safi": "unicast"
        },
        {
          "afi": "ipv6",
          "safi": "unicast"
        }
      ]
    }
  ]
}
```

Maximum Prefix Limit

The BGP Maximum Prefix Limit feature enables you to set a limit for the maximum number of prefixes that a BGP router can receive from its peer router. If a BGP router receives prefixes that exceed the defined limit threshold, the BGP session gets reset and the session goes idle for a pre-defined period.

You can define a period as idle timeout so that the BGP peering gets re-established automatically after the specified time. If you do not specify the idle timeout, the BGP peering does not get re-established until or unless you execute the clear bgp peer command.

Before getting into inactive or idle mode, the router sends a notification message to the peer router about the exceeded threshold with the error code and the sub-code.

You can configure prefix limits for a peer group.

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> **address-family** <afi> <safi> **prefix-limit** <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, or ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast, labeled-unicast, vpn-unicast, or vpn-multicast
count <count>	Number of maximum prefixes that the peer router is allowed to send. The default value is 0. It means no value is configured for prefix limit.
idle-timeout <idle-timeout>	Idle or inactive time after the maximum limit is reached (in minutes). The allowed range is 1 - 2400 min. The default is Forever.

Example: BGP Maximum Prefix Limit Configuration

```
supervisor@L1-STD-2-2002>bm14-tst.fsn.rtbrick.net: cfg> show config instance
default protocol bgp peer-group PE2 address-family ipv4 unicast
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "unicast",
      "prefix-limit": {
        "count": 100,
        "idle-timeout": 5
      }
    }
  ]
}
```

Configure Upstream BGP Route Export into VRFs of Peering Customers

For peering customers, it is required to select and install specific upstream routes in their own dedicated routing tables (VRFs). To achieve this, the routes from a single upstream peer must be distributed to multiple RIBs, one for each customer-specific VRF.

You can configure the Upstream BGP Route Export into VRFs of peering customers.

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> **address-family** <afi> <safi> **export-rib** <name>

Attribute	Description
<instance-name>	Name of the routing instance.
<peer-group-name>	Name of the peer group.
<afi>	Address family identifier (AFI). Supported values: ipv4, or ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast, labeled-unicast, vpn-unicast, or vpn-multicast
export-rib	Defines in which routing tables (RIBs) the routes from this peer will be installed. The option indicates that the routes learned from the BGP peer are to be installed in multiple RIBs.
peer-default-route	It specifies the default route that is to be installed in the customer's routing table.

Example: BGP support for Curated Route Export Configuration

```
supervisor@rtbrick.net: cfg> show config instance default protocol bgp peer-group
PE2 address-family ipv4 unicast
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "unicast",
      "peer-default-route": [
        "CE1",
        "CE2",
        "CE3"
        ],
      "export-rib": [
        {
          "instance": "CE1"
        },
        {
          "instance": "CE2"
        },
        {
          "instance": "CE3"
        }
      ]
    }
  ]
}
```

Peer Configuration

Once peer groups have been defined, BGP peers can be configured at the peer configuration hierarchy. A peer can be specified by address, or by interface when using IPv6 auto-discovered neighbors and link-local addresses. Furthermore, it is possible to configure TCP authentication and bind it to a peer.

Syntax to configure a BGP peer by address:

set instance <instance-name> **protocol bgp peer** (ipv4 | ipv6) <peer-address> <update-source> **peer-group** <peer-group>

Syntax to configure a BGP peer using IPv6 link-local addresses:

set instance <instance-name> **protocol bgp peer interface** <name> **peer-group** <peer-group>

Syntax to configure TCP Authentication for BGP peers:

set instance <instance-name> **protocol bgp peer** (ipv4 | ipv6) <peer-address> <update-source> **authentication-id** <authentication-id>

Attribute	Description
interface <name>	Enable BGP peer using IPv6 link-local addresses
ipv4 <peer-address>	IPv4 address of a BGP peer
ipv6 <peer-address>	IPv6 address of a BGP peer
allow-as-in <value>	Specify the value for allow-as-in. Allowed range of value 1 - 10.

Attribute	Description
<update-source>	Local IP address to be used for the peering
peer-group <peer-group>	Assign the peer to a peer group
deactivate	Deactivate a configured peer
authentication-id <authentication-id>	Authentication identifier

Example 1: BGP peer specified by IP addresses

```
supervisor@rtbrick: cfg> show config instance default protocol bgp peer

{
  "rtbrick-config:peer": {
    "ipv4": [
      {
        "peer-address": "198.51.100.82",
        "update-source": "198.51.100.81",
        "peer-group": "spine"
      }
    ]
  }
}
```

Example 2: BGP peer using IPv6 link-local addresses

```
supervisor@rtbrick: cfg> show config instance default protocol bgp peer

{
  "rtbrick-config:peer": {
    "interface": [
      {
        "name": "ifl-0/0/1/1",
        "peer-group": "spine"
      }
    ]
  }
}
```

Example 3: BGP peer authentication

```
supervisor@rtbrick: cfg> show config instance default protocol bgp peer

{
  "rtbrick-config:peer": {
    "interface": [
      {
        "name": "ifl-0/0/1/1",
        "authentication-id": "auth1",
        "peer-group": "spine"
      }
    ]
  }
}
```

Sample Configuration

Example 1: BGP Configuration of a Spine Switch (Default Instance only)

```
{
  "ietf-restconf:data": {
    "rtbrick-config:instance": [
      {
        "name": "default",
        "ipv4-router-id": "198.51.100.51",
        "protocol": {
          "bgp": {
            "domain-name": "rtbrick.com",
            "host-name": "spine1",
            "local-as": 4200000100,
            "address-family": [
              {
                "afi": "ipv4",
                "safi": "vpn-unicast"
              },
              {
                "afi": "ipv6",
                "safi": "labeled-unicast",
                "srgb": {
                  "base": 5000,
                  "range": 1000,
                  "index": 11
                },
                "redistribute": [
                  {
                    "source": "direct"
                  }
                ]
              },
              {
                "afi": "ipv6",
                "safi": "unicast",
                "redistribute": [
                  {
                    "source": "direct"
                  }
                ]
              },
              {
                "afi": "ipv6",
                "safi": "vpn-unicast"
              }
            ],
            "peer": {
              "interface": [
                {
```

```
                "name": "ifl-0/1/1/1",
                "authentication-id": "auth1",
                "peer-group": "spine"
              },
              {
                "name": "ifl-0/2/1/1",
                "peer-group": "leaf1"
              },
              {
                "name": "ifl-0/2/2/1",
                "peer-group": "leaf2"
              }
            ]
          },
          "peer-group": [
            {
              "pg-name": "leaf1",
              "link-local-nexthop-only": "true",
              "remote-as": 4200000201,
              "address-family": [
                {
                  "afi": "ipv4",
                  "safi": "vpn-unicast",
                  "extended-nexthop": "true",
                  "nexthop-unchanged": "true"
                },
                {
                  "afi": "ipv6",
                  "safi": "labeled-unicast"
                },
                {
                  "afi": "ipv6",
                  "safi": "unicast"
                },
                {
                  "afi": "ipv6",
                  "safi": "vpn-unicast",
                  "nexthop-unchanged": "true"
                }
              ]
            },
            {
              "pg-name": "leaf2",
              "link-local-nexthop-only": "true",
              "remote-as": 4200000202,
              "address-family": [
                {
                  "afi": "ipv4",
                  "safi": "vpn-unicast",
                  "extended-nexthop": "true",
                  "nexthop-unchanged": "true"
                },
                {
                  "afi": "ipv6",
                  "safi": "labeled-unicast"
                },
                {
                  "afi": "ipv6",
                  "safi": "unicast"
                },
                {
                      "afi": "ipv6",
                      "safi": "vpn-unicast",
                      "nexthop-unchanged": "true"
                    }
                  ]
                },
                {
                  "pg-name": "spine",
                  "link-local-nexthop-only": "true",
                  "remote-as": 4200000100,
                  "address-family": [
                    {
                      "afi": "ipv4",
                      "safi": "vpn-unicast",
                      "extended-nexthop": "true"
                    },
                    {
                      "afi": "ipv6",
                      "safi": "labeled-unicast",
                      "nexthop-self": "true"
                    },
                    {
                      "afi": "ipv6",
                      "safi": "unicast",
                      "nexthop-self": "true"
                    },
                    {
                      "afi": "ipv6",
                      "safi": "vpn-unicast"
                    }
                  ]
                }
              ]
            }
          }
        }
      ]
    }
  }
```

Example 2: BGP Configuration of a Leaf Switch with one VPN Instance

```
{
  "ietf-restconf:data": {
    "rtbrick-config:instance": [
      {
        "name": "default",
        "ipv4-router-id": "198.51.100.53",
        "protocol": {
          "bgp": {
            "domain-name": "rtbrick.com",
            "host-name": "leaf1",
            "local-as": 4200000201,
            "address-family": [
              {
                "afi": "ipv4",
                "safi": "vpn-unicast"
              },
              {
                "afi": "ipv6",
                "safi": "labeled-unicast",
                "srgb": {
                  "base": 5000,
                  "range": 1000,
                  "index": 13
                },
                "redistribute": [
                  {
                    "source": "direct"
                  }
                ]
              },
              {
                "afi": "ipv6",
                "safi": "unicast",
                "redistribute": [
                  {
                    "source": "direct"
                  }
                ]
              },
              {
                "afi": "ipv6",
                "safi": "vpn-unicast"
              }
            ],
            "peer": {
              "interface": [
                {
                  "name": "ifl-0/1/1/1"                        "authentication-
id": "auth1",
                  "peer-group": "spine"
                },
                {
                  "name": "ifl-0/1/2/1",
                  "peer-group": "spine"
                }
              ]
            },
            "peer-group": [
              {
                "pg-name": "spine",
                "link-local-nexthop-only": "true",
                "remote-as": 4200000100,
                "address-family": [
                  {
                    "afi": "ipv4",
                    "safi": "vpn-unicast",
                    "extended-nexthop": "true",
                    "update-nexthop": {
                      "ipv6-address": "2001:db8:0:19::"
                    }
                  },
                  {
                    "afi": "ipv6",
                    "safi": "labeled-unicast"
                  },
                  {
                    "afi": "ipv6",
                    "safi": "unicast"
                  },
                  {
                    "afi": "ipv6",
                    "safi": "vpn-unicast",
                    "update-nexthop": {
                      "ipv6-address": "2001:db8:0:19::"
                    }
                  }
                ]
              }
            ]
          }
        }
      },
      {
        "name": "services",
        "ipv4-router-id": "198.51.100.41",
        "route-distinguisher": "198.51.100.41:101",
        "address-family": [
          {
            "afi": "ipv4",
            "safi": "unicast",
            "policy": {
              "export": "MY_V4_POLICY"
            },
            "route-target": {
              "import": "target:198.51.100.70:14",
              "export": "target:198.51.100.70:14"
            }
          },
          {
            "afi": "ipv6",
            "safi": "unicast",
            "policy": {
              "export": "MY_V6_POLICY"
            },
            "route-target": {
              "import": "target:198.51.100.70:16",
              "export": "target:198.51.100.70:16"
            }
          }
        ],
        "protocol": {
          "bgp": {
            "domain-name": "rtbrick.com",
            "host-name": "leaf1",
            "local-as": 65003,
            "address-family": [
              {
                "afi": "ipv4",
                "safi": "unicast",
                "redistribute": [
                  {
                    "source": "direct"
                  },
                  {
                    "source": "ppp"
                  },
                  {
                    "source": "static"
                  }
                ]
              },
              {
                "afi": "ipv6",
                "safi": "unicast",
                "redistribute": [
                  {
                    "source": "direct"
                  },
                  {
                    "source": "ppp"
                  },
                  {
                    "source": "static"
                  }
                ]
              }
            ]
          }
        }
      }
    ]
  }
}
```

2.3.3. BGP Operational Commands

BGP Show Commands

The BGP show commands provide detailed information about the BGP protocol operation and BGP routes.

BGP Summary

This command displays BGP protocol parameters like attributes or timers that are generic to the BGP instance.

Syntax:

show bgp summary <option>

Option	Description
-	Without any option, the commands displays the information for all instances.
instance <instance-name>	BGP summary information for the given instance.

Example: BGP summary for the default instance

```
supervisor@rtbrick: op> show bgp summary instance default
Instance: default
  General information
    Hostname: PE1, Domain name:
    Local AS: 1000, Version: 4
    Local preference: 100, eBGP Protocol preference: 20, iBGP Protocol preference:
200
    Router ID: 198.51.100.102, Cluster ID: 198.51.100.102
  Capabilities
    Route refresh: True, AS4: True, Graceful restart: False, L2VPN EVPN:True L2VPN
VPLS:True
  Best route selection
    Always compare MED: False, Ignore as path: False
    Ignore local preference: False, Ignore origin: False
    Ignore MED: False, Ignore route source: False
    Ignore router ID: False, Ignore uptime: True
    Ignore cluster length: False, Ignore peer IP: False
    Route select parameter: 0
  Timers
    Connect retry: 30s, Keepalive: 30s, Holdtime: 90s
  Statistics
    Peers configured: 1, Peers auto discovery: 0
      Peers in idle          : 0
      Peers in connect       : 0
      Peers in active        : 0
      Peers in opensent      : 0
      Peers in openconfirm   : 0
      Peers in established   : 1
```

To access the Operational State API that corresponds to this CLI, click here.

BGP Peer

The 'show bgp peer' commands display information on BGP peers.

Syntax:

show bgp peer <option> ...

Option	Description
-	Without any option, the commands display all BGP peers in all instances in a summary table format.
detail	Detailed information on all BGP peers in all instances in a list view.
<peer-name>	Detailed information on the peer with the given name.

Option	Description
history	Displays BGP peer history information such as the peer state down reasons.
history <peer-address>	Displays BGP peer history information such as the peer state down reasons for a specified peer.
address <peer-address>	Detailed information on the peer with the given IP address.
instance <instance-name>	Summary of all BGP peers in the given instance.
instance <instance-name> detail	Detailed information on all BGP peers in the given instance.
instance <instance-name> detail <peer-name>	Detailed information on the peer with the given name in the given instance.
instance <instance-name> detail address <peer-address>	Detailed information on the peer with the given IP address in the given instance.
statistics	Received and sent BGP prefixes per AFI/SAFI for all peers in all instances.
statistics peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name.
statistics peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address.
statistics instance <instance-name> peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name in the given instance.
statistics instance <instance-name> peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address in the given instance.

Although 6PE routes are labeled, they are handled as unicast routes, and therefore will be shown as IPv6 unicast in the BGP peer statistics.

Example 1: BGP Peer Summary View

```
supervisor@rtbrick: op> show bgp peer
Instance: default
```

```
  Peer                                     Remote AS    State         Up/Down Time
PfxRcvd              PfxSent
  PE2                                      2000         Established
11d:22h:18m:30s            12                    20
Instance: default
  Peer                                     Remote AS    State         Up/Down Time
PfxRcvd              PfxSent
  CE1                                      65535        Established
6d:02h:28m:02s            2                     2
  CE1                                      65535        Established
6d:02h:27m:45s            2                     2
```

Example 2: BGP Peer Detail View

```
supervisor@rtbrick: op> show bgp peer detail
Peer: PE2, Peer IP: 198.51.100.39, Remote AS: 2000, Local: 198.51.100.29, Local
AS: 1000, Any AS: False
  Type: ebgp, State: Established, Up/Down Time: 11d:22h:18m:48s, Reason: Cease,
Sub-Code: Admin shutdown
  Discovered on interface: -
  Last transition: Thu Nov 19 05:33:28 GMT +0000 2020, Flap count: 1
  Peer ID        : 198.51.100.106, Local ID  : 198.51.100.102
  Instance       : default, Peer group: to_pe2
  6PE enabled    : False
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    Keepalive timer     : expires in 1s 488011us
    Holddown timer      : expires in 1m 15s 85437us
  NLRIs:
    Sent           : ['ipv6-unicast', 'ipv4-vpn-unicast', 'ipv6-vpn-unicast',
'ipv6-labeled-unicast']
    Received       : ['ipv6-unicast', 'ipv6-labeled-unicast', 'ipv4-vpn-unicast',
'ipv6-vpn-unicast']
    Negotiated     : ['ipv6-unicast', 'ipv6-labeled-unicast', 'ipv4-vpn-unicast',
'ipv6-vpn-unicast']
  Capabilities:
    Addpath sent                  : None
    Addpath received              : None
    Addpath negotiated            : None
    Extended nexthop sent         : ['ipv4-vpn-unicast']
    Extended nexthop received     : ['ipv4-vpn-unicast']
    Extended nexthop negotiated   : ['ipv4-vpn-unicast']
    Capabilities:
      Feature                      Sent            Received        Negotiated
      Route refresh                True            True            True
      4 byte AS                    True            True            True
      Graceful restart             False           False           False
      Link local only              False           False           False
  Prefix Limit:
  End of RIB:
    Address family                 Sent                                Received
    IPv4 unicast                   never                               never
    IPv4 labeled-unicast           never                               never
    IPv6 unicast                   Thu Nov 19 05:33:30 GMT +0000 2020  Thu Nov 19
05:33:30 GMT +0000 2020
```

```
    IPv6 labeled-unicast           Thu Nov 19 05:33:30 GMT +0000 2020  Thu Nov 19
05:33:30 GMT +0000 2020
    IPv4 VPN-unicast               Thu Nov 19 05:33:30 GMT +0000 2020  Thu Nov 19
05:33:30 GMT +0000 2020
    IPv6 VPN-unicast               Thu Nov 19 05:33:30 GMT +0000 2020  Thu Nov 19
05:33:30 GMT +0000 2020
    IPv4 VPN-multicast             never                               never
  Message stats:
    Session stats:
      Direction    Open         Update       Keepalive    Notify       Route
refresh
      Input        1            38           41196        0            0
      Output       1            22           41207        0            0
    Total stats:
      Input        2            48           44618        1            0
      Output       3            32           44624        0            0
    Route stats:
      Address family                  Received     Sent
      IPv4 unicast                    0            0
      IPv4 labeled-unicast            0            0
      IPv6 unicast                    2            3
      IPv6 labeled-unicast            2            3
      IPv4 VPN-unicast                4            7
      IPv6 VPN-unicast                4            7
      IPv4 multicast                  0            0
      IPv4 VPN-multicast              0            0
<...>
```

Example 3: BGP Peer Statistics

```
supervisor@rtbrick: op> show bgp peer statistics instance default peer PE2
Instance: default
  Peer           AFI     SAFI                PfxRcvd     PfxSent
  PE2            ipv4    unicast             0           0
                 ipv4    labeled-unicast     0           0
                 ipv6    unicast             2           3
                 ipv6    labeled-unicast     2           3
                 ipv4    vpn-unicast         4           7
                 ipv6    vpn-unicast         4           7
                 ipv4    multicast           0           0
                 ipv4    vpn-multicast       0           0
```

Example 4: BGP Peer history for a specified peer

```
supervisor@rtbrick.net: op> show bgp peer history peer address 192:168::40
Instance: ip2vrf
  Peer Address                 Source Address              Type             Last
Reset Reason
  192:168::40                  192:168:5::20               FSM Error        FSM
Error, Sub-Code: Unexpected message in OpenSent State
```

To access the Operational State API that corresponds to this CLI, click

here.

BGP Peer Group

The 'show bgp peer-group' commands display parameters like BGP attributes that are specific to the respective peer groups.

Syntax:

show bgp peer-group <option> …

Option	Description
-	Without any option, the commands display information on all peer groups in all instances.
<peer-group-name>	Information on the peer group with the given name.
instance <instance-name>	All peer groups in the given instance.
instance <instance-name> <peer-group-name>	Information on the peer group with the given name in the given instance.

Example: BGP Peer Group

```
supervisor@rtbrick: op> show bgp peer-group to_pe2
Instance: default
  Peer group name       : to_pe2
    Remote AS           : 2000
    Import rule         : None
    Export rule         : None
    Remove AS           : None
    Nexthop self        : None
    Multipath iBGP      : None
    Multipath eBGP      : None
    Client-to-Client    : None
    Add path            : None
    eBGP multihop       : None
    Hop (TTL)           : None
    Any AS              : None
    Update VPNv4 NH     : None
    Update MVPN NH      : None
```

BGP FIB

The 'show bgp fib' commands display the BGP forwarding table. In contrast to the 'show bgp rib' commands, the output of the 'show bgp fib' commands includes only the selected routes. The BGP route selection occurs between the RIB and the

FIB.

Syntax:

show bgp fib <option> …

Option	Description
-	Without any option, the commands display the BGP forwarding table for all address families and all instances in a summary table format.
<afi>	BGP forwarding table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> <safi>	BGP forwarding table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'unicast', 'labeled-unicast', 'vpn-multicast', and 'vpn-unicast'.
<afi> <safi> detail	Detailed list view of the BGP forwarding table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP forwarding table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP forwarding table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP forwarding table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP forwarding table entry for the given prefix and instance.

Example 1: Summary view of the BGP FIB for IPv6, all SAFIs and all instances

```
supervisor@rtbrick: op> show bgp fib ipv6
Instance: default, AFI: ipv6, SAFI: unicast
  Prefix                                        Preference      Out Label
Next Hop
  2001:db8:0:2::/32                             20              -
198.51.100.39
  2001:db8:0:2::/32                             20              -
198.51.100.39
Instance: services, AFI: ipv6, SAFI: unicast
```

```
  Prefix                                            Preference      Out Label
Next Hop
  2001:db8:0:6::/32                                 200             -
2001:db8:0:4::
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Prefix                                            Preference      Out Label
Next Hop
  2001:db8:0:2::/32                                 20              2003
198.51.100.39
  2001:db8:0:2::/32                                 20              2003
198.51.100.39
Instance: default, AFI: ipv6, SAFI: vpn-unicast
  Prefix                                            Preference      Out Label
Next Hop
  2001:db8:0:5::/32                                 200             20003,bos:1
  2001:db8:0:6::/32                                 200             20003,bos:1
  2001:db8:0:8::/32                                 200             20003,bos:1
  2001:db8:0:9::/32                                 20              20006,bos:1
2001:db8:0:7::
  2001:db8:0:10::/32                                20              20006,bos:1
2001:db8:0:7::
  2001:db8:0:11::/32                                20              20006,bos:1
2001:db8:0:7::
  2001:db8:0:12::/32                                20              20006,bos:1
2001:db8:0:7::
```

Example 2: Detailed view of the BGP FIB for IPv6 VPN unicast routes in the default instances

```
supervisor@rtbrick: op> show bgp fib ipv6 vpn-unicast instance default detail
Instance: default, AFI: ipv6, SAFI: vpn-unicast
  Prefix: 2001:db8:0:5::/32
    Route source: bgp-local, Send path ID: 405188370, Received path ID: None, Path
hash: None
    AS path: None, Originator ID: None, Origin: Incomplete
    Community: None
    Extended community: ['target:198.51.100.93:2']
    Large community: None
    Cluster list: None
    IGP metric: None, Local preference: 100, Multi exit discriminator: 0
    Preference: 200, External route: None, Readvertised route: None
    Route up: None
    Next hop: 12.0.0.2, Label: 20003,bos:1
  Prefix: 2001:db8:0:6::/32
    Route source: bgp-local, Send path ID: 2400017309, Received path ID: None,
Path hash: None
    AS path: None, Originator ID: None, Origin: Incomplete
    Community: None
    Extended community: ['target:198.51.100.93:2']
    Large community: None
    Cluster list: None
    IGP metric: None, Local preference: 100, Multi exit discriminator: None
    Preference: 200, External route: None, Readvertised route: None
    Route up: None
    Next hop: 12.0.0.2, Label: 20003,bos:1
```

BGP RIB-in

This command displays the total routes.

Syntax:

show bgp rib-in <option> …

Option	Description
-	Without any option, the command displays information on the received BGP routing table on all instances in a summary table format.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'labeled-unicast', 'unicast', 'vpn-multicast', and 'vpn-unicast'.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> community <community-name>	BGP community details for the given AFI, SAFI, and instance.
<afi> <safi> error	BGP route with error status for the given AFI, SAFI, and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Option	Description
import	Summary information for imported routes.

Example 1: Summary view of the BGP rib-in.

```
supervisor@rtbrick: op> show bgp rib-in
Flags: & - Imported, ! - Error
Instance: ip2vrf, AFI: ipv4, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 10
    Prefix                        Next Hop                        MED        Lpref            AS
Path
    198.51.100.75/24              198.51.100.93                       -      100               -
    198.51.100.76/24              198.51.100.94                   -          100               -
    198.51.100.77/24              198.51.100.99                   -          100               -
    198.51.100.78/24              198.51.100.94                   -          100               -
    198.51.100.79/24              198.51.100.99                   -          100               -
    198.51.100.82/24              198.51.100.94                   -          100               -
    198.51.100.93/24              198.51.100.93                       -      100               -
    198.51.100.94/24              198.51.100.94                   -          100               -
    198.51.100.99/24              198.51.100.99                   -          100               -
    198.51.100.99/24              198.51.100.99                   -          100               -
Instance: default, AFI: ipv4, SAFI: vpn-unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 4
    Prefix                        Next Hop                        MED        Lpref             AS
Path
    198.51.100.14/24              2001:db8:0:1::/32               0          -
4200000004
    198.51.100.17/24              2001:db8:0:1::/32               0          -
4200000004
    198.51.100.16/24              2001:db8:0:1::/32               0          -
4200000004
```

Example 2: Summary view of the BGP rib-in for IPv4, with error flag.

```
supervisor@rtbrick>rtbrick.net: op> show bgp rib-in
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 4
    Flags  Prefix           Next Hop    MED     Lpref       AS Path
           12.0.0.0/24      -           0       100         -
           12.1.0.0/24      -           0       100         -
           192.168.0.10/32  -           0       100         -
           192.168.0.11/32  -           0       100         -
  Hostname: P1, Peer IP: 12.0.0.2
  Source IP: 12.0.0.1, Total routes: 4
    Flags  Prefix           Next Hop    MED     Lpref       AS Path
           12.0.0.0/24      12.0.0.2    0       -           4200000002
           12.1.0.0/24      12.0.0.2    0       -           4200000002
           192.168.0.20/32  12.0.0.2    0       -           4200000002
           192.168.0.21/32  12.0.0.2    0       -           4200000002
```

Example 3: Summary view of the BGP rib-in for IPv4, all SAFIs and all instances

```
supervisor@rtbrick: op> show bgp rib-in ipv4
Flags: & - Imported, ! - Error
```

```
Instance: ip2vrf, AFI: ipv4, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 10
    Prefix                 Next Hop                 MED        Lpref              AS Path
    198.51.100.75/24       198.51.100.93             -          100                 -
    198.51.100.76/24       198.51.100.94             -          100                 -
    198.51.100.77/24       198.51.100.95             -          100                 -
    198.51.100.95/24       198.51.100.95             -          100                 -
    198.51.100.99/24        198.51.100.95            -          100                 -
Instance: default, AFI: ipv4, SAFI: vpn-unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 4
    Prefix               Next Hop                   MED        Lpref              AS Path
    198.51.100.14/24      2001:db8:0:13::            0          -                  4200000004
    198.51.100.17/24      2001:db8:0:13::            0          -                  4200000004
```

Example 4: Summary view of the received routes

```
supervisor@rtbrick: op> show bgp rib-in ipv4 unicast instance default peer address 198.51.100.94
Flags: & - Imported, ! - Error
Instance: ip2vrf, AFI: ipv4, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 13
    Prefix                 Next Hop                 MED        Lpref              AS Path
    198.51.100.75/24       198.51.100.93             -          100                 -
    198.51.100.76/24       198.51.100.94             -          100                 -
    198.51.100.77/24       198.51.100.95             -          100
```

BGP RIB-out

This command displays the send routes.

Syntax:

show bgp rib-out <option> ...

Option	Description
-	Without any option, the command displays advertised BGP routes for all instances.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are evpn, evpn-vpws, flowspec, labeled-unicast, multicast, unicast, vpls, vpls-vpws, vpn-multicast, and vpn-unicast.

Option	Description
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the routes advertised to a peer

```
supervisor@rtbrick: op> show bgp rib-out ipv4 unicast peer CE1-Vrf1
Instance: vrf1, AFI: ipv4, SAFI: unicast
  Peer: CE1-Vrf1, Sent routes: 4
    Prefix                              MED             Lpref             Origin          Next Hop
AS Path
    198.51.100.104/24                   0               -                 Incomplete      -
65001
    198.51.100.113/24                   0               -                 Incomplete      -
65001
    198.51.100.117/24                        0               -                 Incomplete      -
65001
    198.51.100.106/24                   0               -                 Incomplete      -
65001
```

Example 2: Detailed view of the routes advertised to a peer

```
supervisor@rtbrick: op> show bgp rib-out
Instance: vrf1, AFI: ipv4, SAFI: unicast
  Peer-group: pe1_ce1, Sent routes: 4
    Prefix                              MED             Lpref             Origin
Next Hop                            AS Path
    198.51.100.104/24                   0               -
Incomplete      -                           65001
    198.51.100.113/24                   0               -
Incomplete      -                           65001
    198.51.100.105/24                   0               -
Incomplete      -                           65001
    198.51.100.106/24                   0               -
Incomplete      -                           65001
Instance: vrf1, AFI: ipv6, SAFI: unicast
  Peer-group: pe1_ce1, Sent routes: 3
```

```
    Prefix                              MED              Lpref             Origin
Next Hop                            AS Path
    2001:db8:0:14::/24                  0                   -
Incomplete      -                                   65001
    2001:db8:0:15::/24                  0                   -
Incomplete      -                                   65001
    2001:db8:0:16::/24                  0                   -
Incomplete      -                                   65001
```

Example 3: Routes advertised to a peer

```
supervisor@rtbrick.net: cfg> show bgp fib ipv4 multicast
Instance: vrf1-blue, AFI: ipv4, SAFI: multicast
  Group           Source                                        Preference
Out Label            Route Type           Next Hop
  -               -                                             20
20002,bos:1          Intra-AS_I-PMSI_AD   12.0.0.1

20002,bos:1          Intra-AS_I-PMSI_AD   12::1
  232.1.1.1/32    10.1.1.1/32                                   20
20002,bos:1          Source_Tree_Join     12.0.0.1

20002,bos:1          Source_Tree_Join     12::1
  232.1.2.1/32    10.1.1.1/32                                   20
20002,bos:1          Source_Tree_Join     12.0.0.1

20002,bos:1          Source_Tree_Join     12::1
```

TCP Connections

This command displays information of the TCP connections used by BGP.

Syntax:

show bgp tcp bgp.iod.1 connection <option> ...

Option	Description
-	Without any option, the command displays the TCP connections used by BGP for all instances.
detail	Detailed list view of the the TCP connections for the given address family (AFI) and sub-address family (SAFI), and all instances.
prefix	TCP connections for the given prefix and all instances.
instance <instance-name>	TCP connections summary for the given instance.

Example 1: Summary view of the BGP TCP connections

```
supervisor@leaf1: cfg> show bgp tcp bgp.iod.1 connection
Instance       Local IP Address                  Remote IP Address           Local
Port  Remote Port  State
default        2001:db8:0:189::                  2001:db8:0:38::             179
49568         Established
default        2001:db8:0:61::                   2001:db8:0:237::            50529
179           Established
```

Example 2: Detailed information of the BGP TCP connections

```
supervisor@leaf1: cfg> show bgp tcp bgp.iod.1 connection detail
Instance: default
  Local IPv6 address       : 2001:db8:0:189::
  Remote IPv6 address      : 2001:db8:0:38::
  Local port               : 179
  Remote port              : 49568
  State                    : Established
    Internal
      Options                  : -- | Keepalive | --
      TOS                      : 0
      TTL                      : 1
      Priority                 : 1
      Flags                    : -|-|-|-|-|Nagle Disabled|-|Wnd Scale|-|-|-
      Last trigger             : 27
      Timer                    : 37624
    Timers
      Poll                     : 0s
      Poll interval            : 0s
      Retransmission           : 65535s
    Receiver
      Expected sequence        : 32965979
      Available window         : 96816
      Announced window         : 95562
      Announced wnd RT edge    : 33061541
      MSS                      : 1440
      RTT estimate             : 0
    Timeout
      Sequence                 : 17683639s
      Retransmission           : 3s
      Retransmissions          : 0s
      Duplicate acks           : 0s
      Highest ack'd sequence   : 17683658s
    Congestion
      Window                   : 162834
      Persist count            : 0
      Send scale               : 5
      Receive scale            : 5
    Sender
      Next seq to send         : 17683658
      Last wnd update seq      : 32965979
      Last wnd update ack      : 17683658
      Window                   : 96192
      Max window announced     : 96800
      Acknowledged             : None
      Send buf                 : 56476
      Send queue length        : 0
      Unsent oversize          : 0
      TS last ack sent         : 818020352
    Keepalive
      Next keepalive idle      : 7200000
      Keepalive interval       : 75000
      Keepalive count          : 9
      Keep sent count          : 0
    Authentication
      Auth type                : HMAC-SHA-256-128
      key1-id                  : 255
      key2-id                  : 0
      Algorithm mismatch       : 0
      Secret mismatch          : 43
      Latest sent digest       : 850fea02c98912ce4497ec2b101a4f7c
      Latest received digest   : a718fb88e0d7fd4a00843e6aec03c864
```

TCP Statistics

This command displays TCP statistics information of the TCP connections used by BGP.

Syntax:

show bgp tcp bgp.iod.1 statistics <option> …

Option	Description
-	Without any option, the command displays the TCP statistics information of the TCP connections used by BGP for all instances.
instance <instance-name>	TCP connections summary for the given instance.

Example: TCP statistics information of the TCP connections used by BGP for the default instance

```
supervisor@rtbrick: op> show bgp tcp bgp.iod.1 statistics instance default
Instance: default
  IP statistics
    Transmitted packets    : 3103242412
    Received packets       : 47351
    Forwarded packets      : 0
    Dropped packets        : 0
    Checksum error         : 0
    Invalid length error   : 0
    Out of memory error    : 0
    Routing error          : 0
    Protocol error         : 0
    Error in options       : 0
    Misc error             : 0
    Cachehit               : 0
  TCP statistics
    Transmitted packets    : 365499779
    Received packets       : 5577
```

```
     Forwarded packets       : 3014656
     Dropped packets         : 46
     Checksum error          : 0
     Invalid length error    : 0
     Out of memory error     : 0
     Routing error           : 3014656
     Protocol error          : 46
     Error in options        : 0
     Misc error              : 2097152
     Cachehit                : 1557594144
```

BGP Clear Commands

Clear commands allow to reset operational states.

BGP Peer

This commands resets BGP peerings.

Syntax:

clear bgp peer <option> ...

Option	Description
all	Clears all the BGP peers.
all soft-in <afi> <safi>	Sends route refresh to all neighbors.
all soft-out <afi> <safi>	Re-advertises all the routes previously sent to the peer.
all stats	Clears the statistics of all the BGP peers.
instance <instance> <peer-ip>	Clears the peer for the given instance and peer IP address.
instance <instance> <peer-ip> source <src-ip>	Clears a specific peer for the given peer IP address and source IP address in the specified instance.
instance <instance> all	Clears all peers in the given instance.
instance <instance> <peer-ip> source <src-ip> soft-in <afi> <safi>	Sends route refresh to specific peer for the given instance, peer-ip, source-ip and address-family.
instance <instance> <peer-ip> soft-in <afi> <safi>	Sends route refresh to peer for the given instance, peer-ip and address-family.

Option	Description
instance <instance> all soft-in <afi> <safi>	Sends route refresh to all peers for the given instance and address family.
instance <instance> <peer-ip> source <src-ip> soft-out <afi> <safi>	Re-advertises all the routes previously sent to the specific peer for the given instance, peer-ip, source-ip and address-family.
instance <instance> <peer-ip> soft-out <afi> <safi>	Sends route refresh to peer for a given instance, peer-ip and address-family.
instance <instance> all soft-out <afi> <safi>	Sends route update to all peers for given instance and address family.
instance <instance> <peer-ip> source <src-ip> stats	Clears the statistics of a specific peer for a given instance, peer-ip and source-IP.
instance <instance> <peer-ip> stats	Clears the statistics of the peer for a given instance and peer-IP.
instance <instance> all stats	Clears the statistics of all peers for a given instance.

Example: The example below shows how to clear all the BGP peers.

```
supervisor@rtbrick: op> clear bgp peer all
```

To access the Operational State API that corresponds to this CLI, click here.

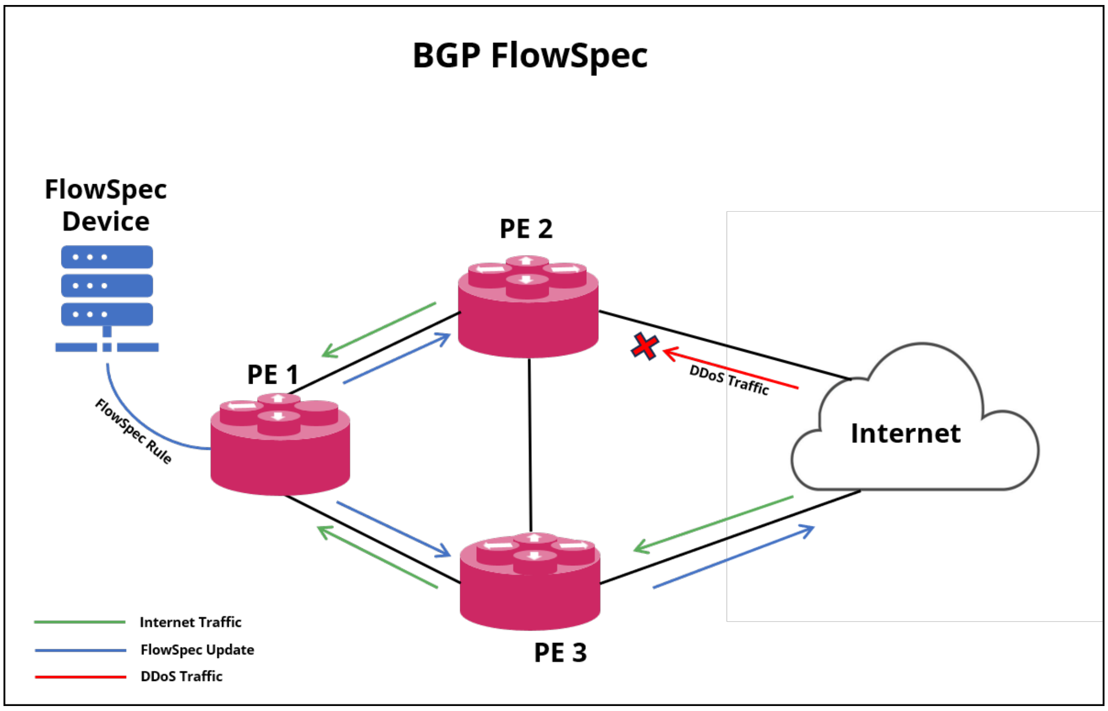
2.4. BGP FlowSpec

2.4.1. BGP FlowSpec Overview

BGP FlowSpec is an extension of the BGP protocol that allows for the dynamic propagation of more specific information than the traffic aggregate defined by an IP Prefix. This enables network administrators to control data traffic flow at any point in their network infrastructure. BGP FlowSpec can be used for various purposes, such as managing congestion or mitigating distributed denial-of-service

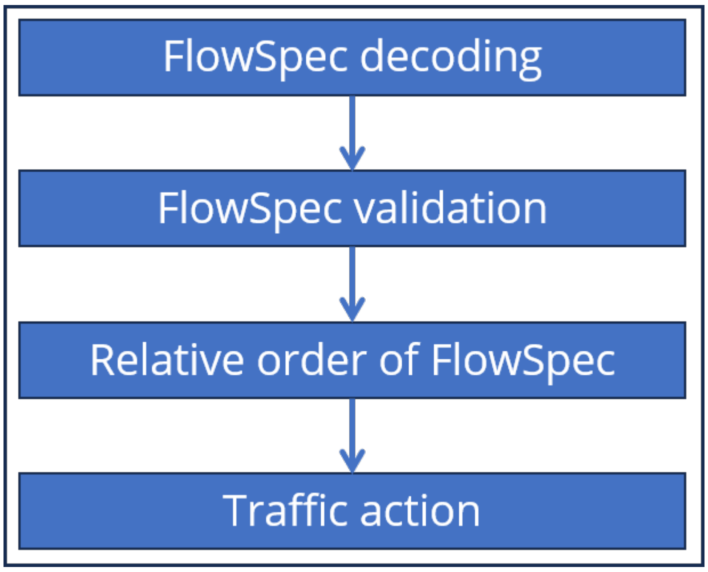
(DDoS) attacks. Expanding routing information with FlowSpec allows the routing system to use the ACL (Access Control List) or firewall capabilities in the router's forwarding path.

The figure below shows a traffic scrubbing station capable of generating the FlowSpec rule in the event of a DDoS attack and sending the BGP FlowSpec update to the neighbouring devices. The device that can interpret this update can drop the DDoS traffic as it arrives.



FlowSpec helps to establish matching criteria for IP traffic packets encoded into BGP Network Layer Reachability Information (NLRI). The requirements can include various attributes and may or may not involve reachability information. Routers can use FlowSpec to forward, shape, classify, rate limit, filter, or redirect packets based on specific policies, allowing for rules that operate on multiple fields of the packet header.

The figure below shows the high-level flow of BGP FlowSpec design.



FlowSpec involves actively processing and inserting dynamic ACLs in an operational environment. When a router receives a FlowSpec update, it can dynamically create IP filters to mitigate intra-AS and inter-AS DDoS attacks and other unwanted traffic patterns. Mitigation is implemented by dropping or rate-limiting the traffic at the network's ingress point (or the nearest possible point toward the source of the DDoS attack).

Supported BGP Standards

RFC Number	Description
RFC 8955	Dissemination of Flow Specification Rules for IPv4
RFC 8956	Dissemination of Flow Specification Rules for IPv6

RFC and draft compliance are partial except as specified.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Supported Matching Criteria and Actions

The tables below outline the FlowSpec supported matching criteria and actions.

BGP FlowSpec NLRI Type	Match Criteria & Description		Option	Supported (Yes/No)
Type 1	Destination-Prefix (ipv4/ipv6)	Defines the destination prefix to match	specific host	Yes
			IP range	Yes
Type 2	Source-Prefix (ipv4/ipv6)	Defines the source prefix to match.	specific host	Yes
			IP range	Yes
Type 3	IP-Protocol	Contains a set of {operator, value} pairs that match the IP protocol value byte in IP packets.	specific value	Yes
			multi-value range	No
Type 4	Port (src or dst)	Defines a list of {operator, value} pairs that match source or destination ports.	specific value	No
			multi-value range	No
Type 5	Destination port	Defines a list of {operator, value} pairs used to match the destination port of a TCP or UDP packet.	specific value	Yes
			multi-value range	No
Type 6	Source port	Defines a list of {operator, value} pairs used to match the source port of a TCP or UDP packet.	specific value	Yes
			multi-value range	No

BGP FlowSpec NLRI Type	Match Criteria & Description		Option	Supported (Yes/No)
Type 7	ICMP type	Defines a list of {operator, value} pairs used to match the type field of an ICMP packet	specific value	No
			multi-value range	No
Type 8	ICMP code	Defines a list of {operator, value} pairs used to match the code field of an ICMP packet.	specific value	No
			multi-value range	No
Type 9	TCP flag	IPv4 or IPv6 TCP flags(2 bytes include reserved bits)	specific value	No
			multi-value range	No
Type 10	Packet length	Match on the total IP packet length	specific value	No
			multi-value range	No
Type 11	DSCP	Defines a list of {operator, value} pairs that use a Multi-value range to match the 6-bit DSCP field.	specific value	No
			multi-value range	No
Type 12	Fragment	Identifies a fragment-type as the match Bit mask criterion for a class map.	specific value	No
			multi-value range	No

The maximum number of matches supported in a single FlowSpec rule is 8.

	Action Criteria & Description		Option	Supported (Yes/No)
1	traffic-rate-bytes	Traffic-rate limits specified in bytes per second	0 (drop)	Yes
			>0	Yes
2	traffic-rate-packets	Traffic-rate limits specified in packets per second	0 (drop)	Yes
			>0	No
3	traffic-action	Action that is performed on the traffic that matches FlowSpec rule	Terminal	No
			Sampling	No
4	rt-redirect	Redirects the traffic to a specific VRF instance or to a next-hop	to vrf	No
			to nexthop	No

Operators Supported in Matching Criteria and Actions

Operator Type	Supported Operators	Unsupported Operators
Relational	equal	greater than, greater than or equal, less than, less than or equal, not equal
Logical	or	and

2.4.2. BGP FlowSpec Configuration

Enabling BGP FlowSpec Address Family

Adding the FlowSpec address family to the BGP address-family configuration enables the exchange of BGP FlowSpec NLRIs.

Syntax:

set <instance> <instance-name> protocol bgp address-family <afi> <safi>

Attribute	Description
<instance-name>	Name of the BGP instance
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6
<safi>	Subsequent address family identifier (SAFI). Specify "flowspec" as the SAFI for BGP to enable the exchange of BGP FlowSpec NLRIs.

Example 1: BGP FlowSpec IPv4 Configuration

```
supervisor@rtbrick>LEAF01: cfg> show config instance default protocol bgp address-
family ipv4 flowspec
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "flowspec"
    }
  ]
}
```

Example 2: BGP FlowSpec IPv6 Configuration

```
supervisor@rtbrick>LEAF01: cfg> show config instance default protocol bgp address-
family ipv6 flowspec
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv6",
      "safi": "flowspec"
    }
  ]
}
```

Enabling BGP FlowSpec in Peer Group Address Family

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6, l2vpn
<safi>	Subsequent address family identifier (SAFI). Specify "flowspec" as the SAFI to enable BGP FlowSpec in Peer Group address family.

Example 1: BGP FlowSpec in Peer Group Address Family

```
supervisor@rtbrick>LEAF01: cfg> show config instance default protocol bgp peer-
group
{
  "rtbrick-config:peer-group": [
    {
      "pg-name": "PE2",
      "remote-as": 4200000002,
      "address-family": [
        {
          "afi": "ipv4",
          "safi": "flowspec"
        },
        {
          "afi": "ipv4",
          "safi": "unicast"
        },
        {
          "afi": "ipv6",
          "safi": "flowspec"
        },
        {
          "afi": "ipv6",
          "safi": "unicast"
        }
      ]
    }
  ]
}
```

2.4.3. BGP FlowSpec Operational Commands

BGP Show Commands

show bgp peer

The 'show bgp peer' commands display information on BGP peers.

Syntax:

show bgp peer <option> …

Option	Description
-	Without any option, the commands display all BGP peers in all instances in a summary table format.
detail	Detailed information on all BGP peers in all instances in a list view.
<peer-name>	Detailed information on the peer with the given name.
history	Displays BGP peer history information such as the peer state down reasons.
history <peer-address>	Displays BGP peer history information such as the peer state down reasons for a specified peer.
address <peer-address>	Detailed information on the peer with the given IP address.
instance <instance-name>	Summary of all BGP peers in the given instance.
instance <instance-name> detail	Detailed information on all BGP peers in the given instance.
instance <instance-name> detail <peer-name>	Detailed information on the peer with the given name in the given instance.
instance <instance-name> detail address <peer-address>	Detailed information on the peer with the given IP address in the given instance.
statistics	Received and sent BGP prefixes per AFI/SAFI for all peers in all instances.
statistics peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name.

Option	Description
statistics peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address.
statistics instance <instance-name> peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name in the given instance.
statistics instance <instance-name> peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address in the given instance

Example 1: BGP Peer Summary View

```
supervisor@rtbrick>LEAF01: op> show bgp peer
Instance: default
  Peer         Remote AS    State         Up/Down Time    PfxRcvd    PfxSent
  PE2          4200000002   Established   0d:01h:35m:53s  42         72
  99.1.1.2     65002        Established   0d:01h:35m:53s  2          27
```

Example 2: BGP Peer Detail View

```
supervisor@rtbrick>LEAF01: op> show bgp peer detail
Peer: PE2, Peer IP: 12.0.0.2, Remote AS: 4200000002, Local: 12.0.0.1, Local AS:
4200000001, Any AS: False
  Type: ebgp, State: Established, Up/Down Time: 0d:01h:38m:59s, Reason: Cease,
Sub-Code: Admin reset
  Discovered on interface: -
  Last transition: Thu Jun 20 09:35:06 GMT +0000 2024, Flap count: 1
  Peer ID        : 192.168.0.20, Local ID  : 192.168.0.10
  Instance       : default, Peer group: PE2
  6PE enabled    : False
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    keepalive timer     : expires in 10s 882996us
    Holddown timer      : expires in 1m 12s 538155us
  NLRIs:
    Sent           : ['l2vpn-evpn', 'l2vpn-vpls', 'ipv4-unicast', 'ipv6-unicast',
'ipv4-flowspec', 'ipv6-flowspec', 'ipv4-vpn-unicast', 'ipv6-vpn-unicast', 'ipv4-
vpn-multicast', 'ipv4-labeled-unicast', 'ipv6-labeled-unicast']
    Received       : ['l2vpn-evpn', 'l2vpn-vpls', 'ipv4-unicast', 'ipv6-unicast',
'ipv4-flowspec', 'ipv6-flowspec', 'ipv4-vpn-unicast', 'ipv6-vpn-unicast', 'ipv4-
vpn-multicast', 'ipv4-labeled-unicast', 'ipv6-labeled-unicast']
    Negotiated     : ['l2vpn-evpn', 'l2vpn-vpls', 'ipv4-unicast', 'ipv6-unicast',
'ipv4-flowspec', 'ipv6-flowspec', 'ipv4-vpn-unicast', 'ipv6-vpn-unicast', 'ipv4-
vpn-multicast', 'ipv4-labeled-unicast', 'ipv6-labeled-unicast']
  Capabilities:
    Addpath sent                  : None
    Addpath received              : None
    Addpath negotiated            : None
    Extended nexthop sent         : None
    Extended nexthop received     : None
    Extended nexthop negotiated   : None
    Capabilities:
      Feature                       Sent            Received        Negotiated
      Route refresh                 True            True            True
      4 byte AS                     True            True            True
      Graceful restart              False           False           False
      Link local only               False           False           False
  Prefix Limit:
  End of RIB:
    Address family                  Sent                                Received
    IPv4 unicast                    Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv4 labeled-unicast            Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv6 unicast                    Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv6 labeled-unicast            Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv4 VPN-unicast                Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv6 VPN-unicast                Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv4 flowspec                   Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv6 flowspec                   Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    IPv4 VPN-multicast              Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    L2VPN VPLS                      Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
    L2VPN EVPN                      Thu Jun 20 09:35:11 GMT +0000 2024  Thu Jun 20
09:35:11 GMT +0000 2024
  Message stats:
    Session stats:
      Direction   Open          Update        Keepalive     Notify        Route
refresh
      Input       1             40            235           0             0
      Output      1             52            239           0             0
    Total stats:
      Input       2             80            299           0             0
      Output      2             102           302           1             0
    Route stats:
      Address family                  Received      Sent          Prefix limit Idle
timeout
      IPv4 unicast                    4             4             0             0
      IPv4 labeled-unicast            2             2             0             0
      IPv6 unicast                    4             4             0             0
      IPv6 labeled-unicast            2             2             0             0
      IPv4 VPN-unicast                8             4             0             0
      IPv6 VPN-unicast                6             4             0             0
      IPv4 VPN-multicast              2             6             0             0
      L2VPN VPLS                      7             7             0             0
      L2VPN EVPN                      7             7             0             0
      IPv4 flowspec                   0             8             0             0
      IPv6 flowspec                   0             0             0             0
Peer: , Peer IP: 99.1.1.2, Remote AS: 65002, Local: 99.1.1.1, Local AS:
4200000001, Any AS: False
```

```
  Type: ebgp, State: Established, Up/Down Time: 0d:01h:38m:59s, Reason: Cease,
Sub-Code: Admin reset
  Discovered on interface: -
  Last transition: Thu Jun 20 09:35:06 GMT +0000 2024, Flap count: 2
  Peer ID        : 192.168.1.3, Local ID  : 192.168.0.10
  Instance       : default, Peer group: SN
  6PE enabled    : False
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    keepalive timer     : expires in 14s 885374us
    Holddown timer      : expires in 1m 806199us
  NLRIs:
    Sent           : ['ipv4-unicast', 'ipv6-unicast', 'ipv4-flowspec', 'ipv6-
flowspec']
    Received       : ['ipv4-flowspec', 'ipv6-flowspec']
    Negotiated     : ['ipv4-flowspec', 'ipv6-flowspec']
  Capabilities:
    Addpath sent                  : None
    Addpath received              : None
    Addpath negotiated            : None
    Extended nexthop sent         : None
    Extended nexthop received     : ['ipv4-flowspec', 'ipv6-flowspec']
    Extended nexthop negotiated   : None
    Capabilities:
      Feature                      Sent            Received        Negotiated
      Route refresh                True            True            True
      4 byte AS                    True            True            True
      Graceful restart             False           False           False
      Link local only              False           False           False
<...>
```

Example 4: BGP Peer history for a specified peer

```
supervisor@rtbrick.net: op> show bgp peer history peer address 192:168::40
Instance: ip2vrf
  Peer Address                Source Address             Type             Last
Reset Reason
  192:168::40                 192:168:5::20              FSM Error        FSM
Error, Sub-Code: Unexpected message in OpenSent State
```

show bgp rib-in

This command displays the received routes.

Syntax:

show bgp rib-in <option> ...

Option	Description
-	Without any option, the command displays information on the received BGP routing table on all instances in a summary table format.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'labeled-unicast', 'unicast', 'vpn-multicast', 'vpn-unicast', ‘evpn-vpws’, ‘evpn’, ‘vpls-vpws’, ‘vpls’, and ‘flowspec’.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> community <community-name>	BGP community details for the given AFI, SAFI, and instance.
<afi> <safi> error	BGP route with error status for the given AFI, SAFI, and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the BGP rib-in for the ipv4 flowspec address family.

```
supervisor@rtbrick>LEAF01: op> show bgp rib-in ipv4 flowspec
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
Peer IP: 99.1.1.2, Source IP: 99.1.1.1, Total routes: 2
   Flowspec Hash        Match                                                Action
```

```
AS Path       Status
   236e3111             src-prefix  : 192.0.2.3/32                        rate-
limit:400.0 kbps   65002         Valid
                        ip-proto    : [ ==tcp or ==udp ]
                        src-port    : [ ==200 or ==100 or ==300 ]
   e05a9523             dest-prefix : 203.0.113.0/24                      discard
65002         Valid
```

Example 2: Summary view of the BGP rib-in for the ipv6 flowspec address family.

```
supervisor@rtbrick>LEAF01: op> show bgp rib-in ipv6 flowspec
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv6, SAFI: flowspec
Peer IP: 99.1.1.2, Source IP: 99.1.1.1, Total routes: 1
   Flowspec Hash                Match
Action                    AS Path       Status
   eff682bf                     src-prefix  : 2001:db8::1/128
rate-limit:500.0 kbps     -             Valid
                                ip-proto    : [ ==udp ]
                                src-port    : [ ==4000 or ==5000 ]
```

Example 3: Summary view of the BGP rib-in for the IPv4 with the error flag.

```
supervisor@rtbrick>rtbrick.net: op> show bgp rib-in ipv4
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 4
    Flags  Prefix           Next Hop  MED       Lpref       AS Path
           12.0.0.0/24      -         0         100         -
           12.1.0.0/24      -         0         100         -
           192.168.0.10/32  -         0         100         -
           192.168.0.11/32  -         0         100         -
  Hostname: P1, Peer IP: 12.0.0.2
  Source IP: 12.0.0.1, Total routes: 4
    Flags  Prefix           Next Hop  MED       Lpref       AS Path
           12.0.0.0/24      12.0.0.2  0         -           4200000002
           12.1.0.0/24      12.0.0.2  0         -           4200000002
           192.168.0.20/32  12.0.0.2  0         -           4200000002
           192.168.0.21/32  12.0.0.2  0         -           4200000002
```

show bgp rib-out

This command displays the send routes.

Syntax:

show bgp rib-out <option> ...

Option	Description
-	Without any option, the command displays advertised BGP routes for all instances.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'unicast', 'labeled-unicast', 'multicast', 'vpn-unicast', ‘evpn’, ‘vpls’, ‘vpls-vpws’, and ‘flowspec’.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the IPv4 FlowSpec routes advertised to a peer

```
supervisor@rtbrick>LEAF01: op> show bgp rib-out ipv4 flowspec
Instance: default, AFI: ipv4, SAFI: flowspec
  Peer-group: PE2, Sent routes: 2
   Flowspec Hash                Match
Action                       Origin
   236e3111                     src-prefix  : 192.0.2.3/32
rate-limit:400.0 kbps        Incomplete
                                ip-proto    : [ ==tcp or ==udp ]
                                src-port    : [ ==200 or ==100 or ==300 ]
   e05a9523                     dest-prefix : 203.0.113.0/24
discard                      Incomplete
```

Example 2: Summary view of the IPv6 FlowSpec routes advertised to a peer

```
supervisor@rtbrick>LEAF01: op> show bgp rib-out ipv6 flowspec
Instance: default, AFI: ipv6, SAFI: flowspec
  Peer-group: PE2, Sent routes: 1
   Flowspec Hash                Match
Action                       Origin
   eff682bf                     src-prefix  : 2001:db8::1/128
rate-limit:500.0 kbps        Incomplete
                                ip-proto    : [ ==udp ]
                                src-port    : [ ==4000 or ==5000 ]
```

show bgp fib

The 'show bgp fib' commands display the BGP forwarding table. In contrast to the 'show bgp rib' commands, the output of the 'show bgp fib' commands includes only the selected routes. The BGP route selection occurs between the RIB and the FIB.

Syntax:

show bgp fib <option> ...

Option	Description
-	Without any option, the commands display the BGP forwarding table for all address families and all instances in a summary table format.
<afi>	BGP forwarding table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> <safi>	BGP forwarding table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'unicast', 'labeled-unicast', 'vpn-multicast', 'vpn-unicast', ‘evpn-vpws’, ‘vpls’, ‘vpls-vpws’ and ‘flowspec’.
<afi> <safi> detail	Detailed list view of the BGP forwarding table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP forwarding table entry for the given prefix and all instances.

Option	Description
<afi> <safi> instance <instance-name>	BGP forwarding table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP forwarding table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP forwarding table entry for the given prefix and instance.

Example 1: Summary view of the BGP FIB for IPv4 flowspec address family

```
supervisor@rtbrick>LEAF01: op> show bgp fib ipv4 flowspec
Instance: default, AFI: ipv4, SAFI: flowspec
  Flowspec Hash             Match                                     Action
Priority   Status
  236e3111                  src-prefix  : 192.0.2.3/32                rate-
limit:400.0 kbps     1502       Installed
                            ip-proto    : [ ==tcp or ==udp ]
                            src-port    : [ ==200 or ==100 or ==300 ]
  e05a9523                  dest-prefix : 203.0.113.0/24              discard
1501       Installed
```

Example 2: Summary view of the BGP FIB for a specific IPv4 flowspec hash

```
supervisor@rtbrick>LEAF01: op> show bgp fib ipv4 flowspec e05a9523
Instance: default, AFI: ipv4, SAFI: flowspec
  Flowspec hash: e05a9523
    Match:
      dest-prefix:  203.0.113.0/24
    Action:
      discard
    Extended community:
      flowspec:traffic-rate-bytes:0:0.000000
    Priority: 1509
    Status: Installed
    Number of ACL installed: 1
      Rule: bgp-flowspec-e05a95230d1f7153ac561b564947a9457b1083f57fa5cb10
        ACL type: l3v4
        Ordinal: 0               Priority: 1501
        Match:
          Destination IPv4 prefix: 203.0.113.0/24
        Action:
          Drop: True
        Statistics:
                    Total         Accepted      Dropped
          Packets:  45            0             45
          Bytes:    11700         0             11700
```

Example 3: Summary view of the BGP FIB for IPv6 flowspec address family

```
supervisor@rtbrick>LEAF01: op> show bgp fib ipv6 flowspec
Instance: default, AFI: ipv6, SAFI: flowspec
  Flowspec Hash             Match                                     Action
Priority   Status
  eff682bf                  src-prefix  : 2001:db8::1/128             rate-
limit:500.0 kbps     1501       Installed
                            ip-proto    : [ ==udp ]
                            src-port    : [ ==4000 or ==5000 ]
```

Example 3: Summary view of the BGP FIB for a specific IPv4 flowspec address family

```
supervisor@rtbrick>LEAF01: op> show bgp fib ipv4 flowspec 236e3111
Instance: default, AFI: ipv4, SAFI: flowspec
  Flowspec hash: 236e3111
    Match:
      src-prefix:  192.0.2.3/32
      ip-proto:  [ ==tcp or ==udp ]
      src-port:  [ ==200 or ==100 or ==300 ]
    Action:
      rate-limit:400.0 kbps
    Extended community:
      flowspec:traffic-rate-bytes:0:400000.000000
    Priority: 1502
    Status: Installed
    Number of ACL installed: 6
      Rule: bgp-flowspec-236e31110de8e5920e5caed1e4bc3fe92d570bda99bd49f7
        ACL type: l3v4
        Ordinal: 4               Priority: 1502
        Match:
          Source IPv4 prefix: 192.0.2.3/32
          Source L4 port:: 100
          IP protocol: udp
        ACL type: l3v4
        Ordinal: 3               Priority: 1502
        Match:
          Source IPv4 prefix: 192.0.2.3/32
          Source L4 port:: 200
          IP protocol: udp
        ACL type: l3v4
        Ordinal: 0               Priority: 1502
        Match:
          Source IPv4 prefix: 192.0.2.3/32
          Source L4 port:: 200
          IP protocol: tcp
        ACL type: l3v4
        Ordinal: 2               Priority: 1502
        Match:
          Source IPv4 prefix: 192.0.2.3/32
          Source L4 port:: 300
          IP protocol: tcp
        ACL type: l3v4
        Ordinal: 5               Priority: 1502
<...>
```

Show Command Filter Options for RIB-in, RIB-out, and FIB

Syntax

show bgp fib|rib-in|rib-out <afi> **flowspec filter** <options>

Option	Description
<afi> flowspec filter <options>	Filter BGP Flowspec entries across the RIB based on the option specified.
destination-port	Filters based on destination port number.
destination-prefix	Filters based on the destination IP prefix.
instance-name	Filters entries based on the BGP instance. Useful to verify the Flowspec entries specific to one instance in a multi-instance BGP configurations.
ip-proto	Filters based on the protocol.
port	Filters Flowspec entries based on the protocol port number.
source-port	Filters entries based on the source port. Useful to identify different applications or services originating from specific ports.
source-prefix	Filters entries based on the source IP prefix.

```
supervisor@rtbrick.net: cfg> show bgp rib-in ipv4 flowspec  filter  port 3344
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
  Hostname: SN, Peer IP: 23.1.1.3
  Source IP: 23.1.1.1, Total routes: 9999
    Flags  Flowspec Hash                               Match
Action                        AS Path
           647718a2                                    dest-prefix : 10.1.1.1/32
None                          65002
                                                       src-prefix  :
123.123.133.1/32
                                                       ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                       port        : [ ==3344 and
>=3345 or >3346 ]
                                                       dst-port    : [ ==3344 and
>=3345 or >3346 ]
                                                       src-port    : [ ==3344 and
>=3345 or >3346 ]
           f547079c                                    dest-prefix : 10.1.1.1/32
None                          65002
                                                       src-prefix  :
123.123.133.1/32
```

```
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==3343 and
>=3344 or >3345 ]
                                                        dst-port    : [ ==3343 and
>=3344 or >3345 ]
                                                        src-port    : [ ==3343 and
>=3344 or >3345 ]
           fa25370c                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==3342 and
>=3343 or >3344 ]
                                                        dst-port    : [ ==3342 and
>=3343 or >3344 ]
                                                        src-port    : [ ==3342 and
>=3343 or >3344 ]
```

```
supervisor@rtbrick.net: cfg> show bgp rib-in ipv4 flowspec filter destination-
prefix 20.1.1.1/32
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
  Hostname: SN, Peer IP: 23.1.1.3
  Source IP: 23.1.1.1, Total routes: 9999
    Flags  Flowspec Hash                                Match
Action                        AS Path
           36f587c2                                     dest-prefix : 20.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==28 and
>=29 or >30 ]
                                                        dst-port    : [ ==28 and
>=29 or >30 ]
                                                        src-port    : [ ==28 and
>=29 or >30 ]
           dbedeed9                                     dest-prefix : 20.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==9978 and
>=9979 or >9980 ]
                                                        dst-port    : [ ==9978 and
>=9979 or >9980 ]
                                                        src-port    : [ ==9978 and
>=9979 or >9980 ]
```

```
supervisor@rtbrick.net: cfg> show bgp rib-in ipv4 flowspec filter source-prefix
20.123.133.1/32
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
```

```
  Hostname: SN, Peer IP: 23.1.1.3
  Source IP: 23.1.1.1, Total routes: 9999
    Flags  Flowspec Hash                                Match
Action                        AS Path
           38bfcfd3                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
20.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==251 and
>=252 or >253 ]
                                                        dst-port    : [ ==251 and
>=252 or >253 ]
                                                        src-port    : [ ==251 and
>=252 or >253 ]
           5fa337e5                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
20.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==9997 and
>=9998 or >9999 ]
                                                        dst-port    : [ ==9997 and
>=9998 or >9999 ]
                                                        src-port    : [ ==9997 and
>=9998 or >9999 ]
           91c84623                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
20.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==154 and
>=155 or >156 ]
                                                        dst-port    : [ ==154 and
>=155 or >156 ]
                                                        src-port    : [ ==154 and
>=155 or >156 ]
```

```
supervisor@rtbrick.net: cfg> show bgp rib-in ipv4 flowspec filter destination-port
100
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
  Hostname: SN, Peer IP: 23.1.1.3
  Source IP: 23.1.1.1, Total routes: 9999
    Flags  Flowspec Hash                                Match
Action                        AS Path
           14c614cf                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==100 and
>=101 or >102 ]
                                                        dst-port    : [ ==100 and
>=101 or >102 ]
```

```
                                                        src-port    : [ ==100 and
>=101 or >102 ]
           2eaa5fab                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==99 and
>=100 or >101 ]
                                                        dst-port    : [ ==99 and
>=100 or >101 ]
                                                        src-port    : [ ==99 and
>=100 or >101 ]
           3b8548f9                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==98 and
>=99 or >100 ]
                                                        dst-port    : [ ==98 and
>=99 or >100 ]
                                                        src-port    : [ ==98 and
>=99 or >100 ]
```

```
supervisor@rtbrick.net: cfg> show bgp rib-in ipv4 flowspec filter source-port  200
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
  Hostname: SN, Peer IP: 23.1.1.3
  Source IP: 23.1.1.1, Total routes: 9999
    Flags  Flowspec Hash                                Match
Action                        AS Path
           908f5353                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==199 and
>=200 or >201 ]
                                                        dst-port    : [ ==199 and
>=200 or >201 ]
                                                        src-port    : [ ==199 and
>=200 or >201 ]
           a6c7b490                                     dest-prefix : 10.1.1.1/32
None                          65002
                                                        src-prefix  :
123.123.133.1/32
                                                        ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                        port        : [ ==200 and
>=201 or >202 ]
                                                        dst-port    : [ ==200 and
>=201 or >202 ]
                                                        src-port    : [ ==200 and
>=201 or >202 ]
           c2a0d3e2                                     dest-prefix : 10.1.1.1/32
```

```
None                              65002
                                                                src-prefix  :
123.123.133.1/32
                                                                ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp ]
                                                                port        : [ ==198 and
>=199 or >200 ]
                                                                dst-port    : [ ==198 and
>=199 or >200 ]
                                                                src-port    : [ ==198 and
>=199 or >200 ]
```

```
supervisor@rtbrick.net: cfg> show bgp rib-in ipv4 flowspec filter ip-proto icmp
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: flowspec
  Hostname: SN, Peer IP: 23.1.1.3
  Source IP: 23.1.1.1, Total routes: 9999
    Flags  Flowspec Hash                                        Match
Action                            AS Path
           3dca36d8                                             dest-prefix : 10.1.1.1/32
None                              65002
                                                                src-prefix  :
123.123.133.1/32
                                                                ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp or <icmp ]
                                                                port        : [ ==9516 and
>=9517 or >9518 ]
                                                                dst-port    : [ ==9516 and
>=9517 or >9518 ]
                                                                src-port    : [ ==9516 and
>=9517 or >9518 ]
           db68d8a6                                             dest-prefix : 10.1.1.1/32
None                              65002
                                                                src-prefix  :
123.123.133.1/32
                                                                ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp or >icmp ]
                                                                port        : [ ==29 and
>=30 or >31 ]
                                                                dst-port    : [ ==29 and
>=30 or >31 ]
                                                                src-port    : [ ==29 and
>=30 or >31 ]
           deb2048d                                             dest-prefix : 10.1.1.1/32
None                              65002
                                                                src-prefix  :
123.123.133.1/32
                                                                ip-proto    : [ ==tcp or
!=ospf or >=udp or <=igmp or <icmp ]
                                                                port        : [ ==9674 and
>=9675 or >9676 ]
                                                                dst-port    : [ ==9674 and
>=9675 or >9676 ]
                                                                src-port    : [ ==9674 and
>=9675 or >9676 ]
```

Validating FlowSpec ACLs

In forwarding, the FlowSpec rules can be validated using the "show acl rule <…>" command as shown in the example below:

```
supervisor@rtbrick>LEAF01: op> show acl rule bgp-flowspec-
b9342ffc0d1f7153ac561b564947a9457b1083f57fa5cb10
Rule: bgp-flowspec-b9342ffc0d1f7153ac561b564947a9457b1083f57fa5cb10
  ACL type: l3v4
  Ordinal: 0             Priority: 1501
    Match:
      Direction: ingress
      Destination IPv4 prefix: 203.0.113.0/24
    Action:
      Stats enabled: True
      Drop: True
    Result:
      ACL Handle: 93
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    45          0           45
      Bytes      11700       0           11700
```

BGP FlowSpec Clear Commands

Clear commands allow to reset operational states.

BGP Peer

This commands resets BGP peerings.

Syntax:

clear bgp peer <option> …

Option	Description
all	Clears all the BGP peers.
all soft-in <afi> <safi>	Sends route refresh to all neighbors.
all soft-out <afi> <safi>	Re-advertises all the routes previously sent to the peer.
all stats	Clears the statistics of all the BGP peers.
instance <instance> <peer-ip>	Clears the peer for the given instance and peer IP address.

Option	Description
instance <instance> <peer-ip> source <src-ip>	Clears a specific peer for the given peer IP address and source IP address in the specified instance.
instance <instance> all	Clears all peers in the given instance.
instance <instance> <peer-ip> source <src-ip> soft-in <afi> <safi>	Sends route refresh to specific peer for the given instance, peer-ip, source-ip and address-family.
instance <instance> <peer-ip> soft-in <afi> <safi>	Sends route refresh to peer for the given instance, peer-ip and address-family.
instance <instance> all soft-in <afi> <safi>	Sends route refresh to all peers for the given instance and address family.
instance <instance> <peer-ip> source <src-ip> soft-out <afi> <safi>	Re-advertises all the routes previously sent to the specific peer for the given instance, peer-ip, source-ip and address-family.
instance <instance> <peer-ip> soft-out <afi> <safi>	Sends route refresh to peer for a given instance, peer-ip and address-family.
instance <instance> all soft-out <afi> <safi>	Sends route update to all peers for given instance and address family.
instance <instance> <peer-ip> source <src-ip> stats	Clears the statistics of a specific peer for a given instance, peer-ip and source-IP.
instance <instance> <peer-ip> stats	Clears the statistics of the peer for a given instance and peer-IP.
instance <instance> all stats	Clears the statistics of all peers for a given instance.

Example: The example below shows how to clear all the BGP peers.

```
supervisor@rtbrick: op> clear bgp peer all
```

Example: Route Refresh in IPv4/IPv6 for BGP FlowSpec

```
supervisor@rtbrick: op> clear bgp peer all soft-out ipv4 flowspec
```

```
supervisor@rtbrick: op> clear bgp peer all soft-in ipv6 flowspec
```

2.5. BGP RPKI-RTR Route Validation

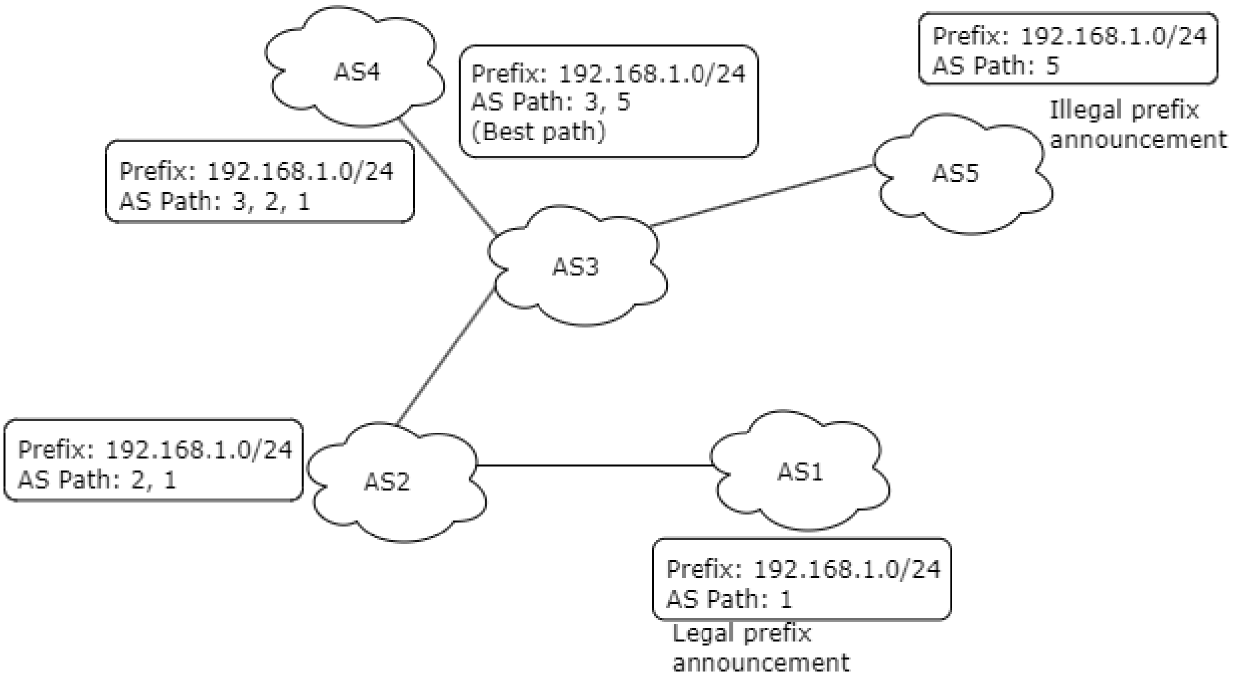
2.5.1. BGP RPKI-RTR Prefix Validation Overview

BGP is the core of the Internet, serving as the backbone protocol that enables exchange of traffic between networks. It plays a crucial role in connecting autonomous systems and ensuring reliable data transmission across the Internet. However, BGP has some inherent security limitations due to its design. It cannot verify whether incoming BGP advertisements come from an authorized autonomous system (AS). This limitation leaves BGP open to various attacks such as route hijacking.

Challenges Due to False Prefix Announcements

Route Hijacking

An attacker can falsely advertise ownership of an IP prefix to redirect or intercept traffic. Also, an autonomous system can incorrectly advertise routes it learned from one provider to other providers. This can be an accidental route leak that directs routes to unintended destinations.



The preceding image is an example of a prefix hijacking scenario. ‘AS1’ announces

192.168.1.0/24, while an attacker announces the same prefix from ‘AS5’. ‘AS1’ is the only authorized to originate and announce this prefix on the Internet. ‘AS5’, without authorization, sends a BGP announcement for the same prefix, 192.168.1.0/24. As a result, other ASes on the Internet receive two conflicting announcements for this prefix, one from 'AS1' and another from 'AS5'. AS4 prefers AS5’s announcement due to the shortest AS path. So, AS5’s unauthorized announcement is selected as the best path; traffic intended for 192.168.1.0/24 will be rerouted to 'AS5' instead of 'AS1'.

Route Hijacking occurs in a number of ways:

Prefix Hijacking: An AS can advertise IP prefixes that it does not legitimately own. Attacker can intercept traffic for those prefixes.

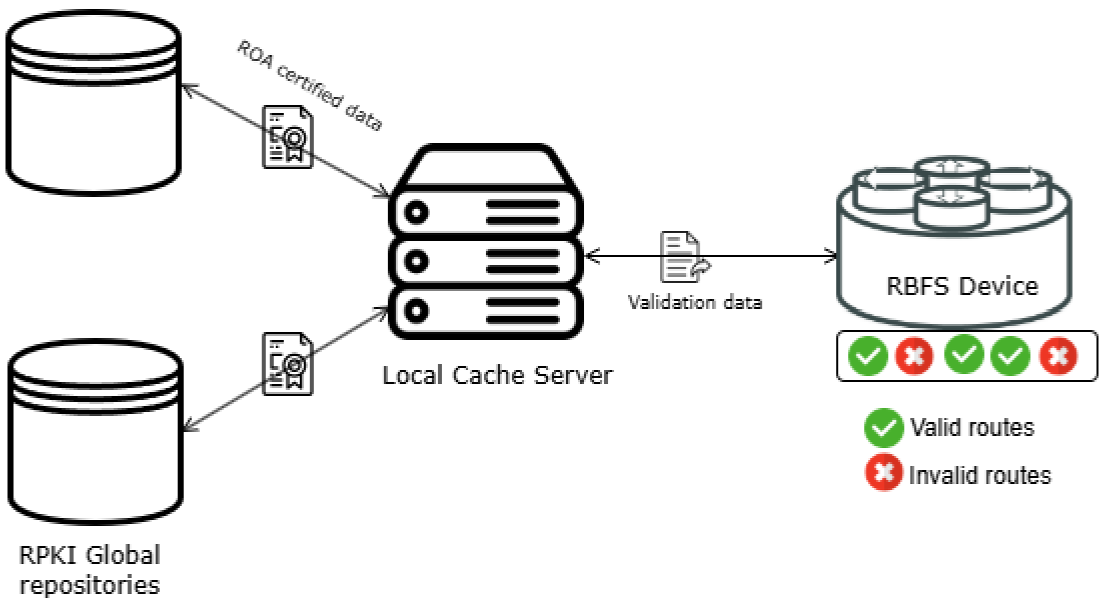
AS Path Hijacking: In this scenario, an AS modifies the AS path of a route advertisement to attract traffic that would normally pass through another AS. This can lead to malicious traffic interception or disruption.

Misconfiguration: A simple configuration error can cause large-scale traffic redirection that affects the internet connectivity.

RBFS BGP implements RPKI-RTR based prefix validation mechanism that verifies the authenticity of route announcements before they are accepted and propagated.

Understanding BGP RPKI-RTR Validation Implementation

Resource Public Key Infrastructure (RPKI) is a security framework to protect BGP routes from specific types of attacks like route hijacking and misconfigurations. It provides a way to verify the origin AS of BGP routes, ensuring that only authorized ASs are allowed to advertise specific IP prefixes. It achieves this by cryptographically associating IP address allocations (both IPv4 and IPv6 prefixes) with their rightful owner ISPs or enterprises.



Key Components of the Validation Mechanism

Global RPKI Repository

Global RPKI servers are maintained by Regional Internet Registries and other trusted entities. These servers contain cryptographic certificates and Route Origin Authorizations (ROAs) for IP prefixes and ASNs. This is the authoritative source where the reliable information about IP address ownership.

Local Cache Server

ISPs deploy local cache servers, which are configured to regularly synchronize with global RPKI repositories, mirroring the latest ROAs and certified data. The local cache servers serve as intermediaries between the global repository and RBFS devices.

RBFS Device

Once configured, RBFS device establishes secure communication channel with the local cache server for fetching the data using the RTR protocol. Once fetched, the RPKI data is stored locally within the RBFS system. Data are stored in a data table with entries that correspond to valid route origin authorizations. It allows quick lookups for route validation when the device receives a new prefix announcement from a BGP peer.

How RPKI Validation Works

- RPKI Data Repos: Network operators create ROAs which cryptographically bind an IP prefix with an authorized ASN. ROAs are stored in global RPKI repositories which are trusted sources are accessible globally. Local caches servers access the data and stores the verified data.
- RPKI Data: RBFS retrieves of data from cache servers. After establishing a validation session with a cache server, RBFS periodically sends reset queries at refresh intervals. These queries request the complete set of records from the cache. When RBFS receives an end-of-data (EOD) PDU, it performs garbage collection to remove outdated records. RBFS maintains the verified data from the local cache servers and stores the data within the system.
- Route Announcement: When a new prefix is advertised from a peer, RBFS receives the advertisement.
- Validation: RBFS looks up and verifies the ROA to see if the origin AS in the BGP route matches the authorized AS in the ROA using data stored in its RPKI data table. The result of this verification can be 'Valid', 'Invalid', or 'Unknown'.
- Decision: Based on the configured validation filter ('strict', 'loose', or 'disable'), the RBFS decides whether to accept or reject the route.

Once the route is validated, RBFS follows decision process as follows:

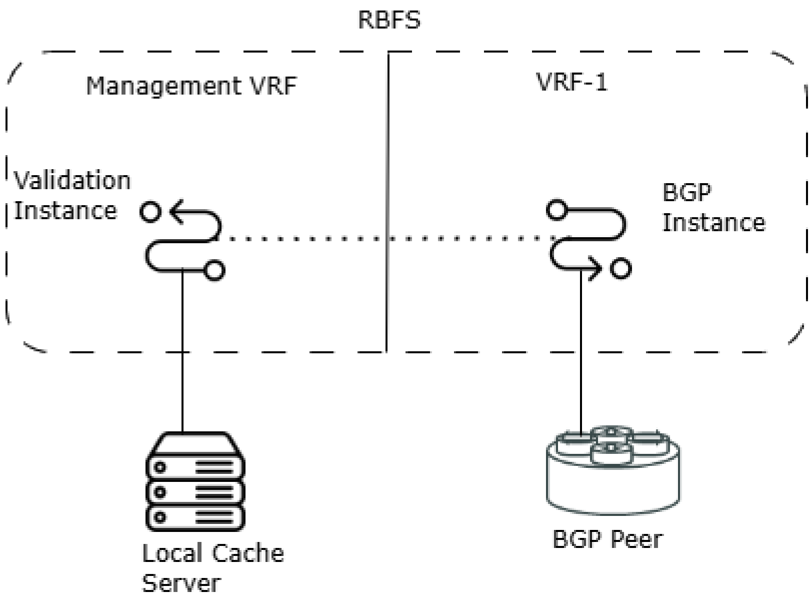
- Valid Routes: Valid routes are installed into the routing table and used for traffic forwarding.
- Invalid Routes: Invalid routes are discarded.
- Unknown: Routes that are not found in the RPKI data will be accepted if you have configured the filter option as loose. Otherwise, this will be discarded.

This entire process ensures that only verified and authorized routes are propagated within the routing tables.

IP prefixes may change any time. Once configured, an RBFS device continuously validates prefixes in real-time when new BGP route announcements arrive. If the RPKI data changes (for example, a new ROA is added or updated), the RBFS device needs to revalidate routes which were accepted or rejected previously. This ensures that only routes verified through RPKI are accepted and malicious route announcements are rejected.

Shared Validation Databases

RBFS allows BGP prefixes in one instance to use the validation database from another VRF instance. It allows for sharing the validation database between routing instances within the system. When a prefix is received in a BGP update, the receiving BGP instance checks its validity by querying the shared database.



Timers

With timers, RBFS ensures that the router maintains an up-to-date RPKI validation database. Timers are used to periodically refresh, retry, and expire validation data based on these intervals. RBFS uses timers to maintain synchronization between local cache and RBFS device.

Refresh Interval: Specifies how frequently the router should query the Local Cache to refresh its RPKI validation database. Router polls the cache with Reset Query PDU at Refresh intervals.

Retry Interval: Specifies how long the router should wait before retrying after a failed Reset Query PDU. Router sends Reset Query PDU upon the failed reset query at retry interval.

Expire Interval: Defines how long the router can continue using the current version of the validation database if communication with the Local Cache fails. Router uses the current version of validation database until expire interval.

RBFS clears the validation session and the database after three consecutive unsuccessful reset query attempts at the refresh interval.

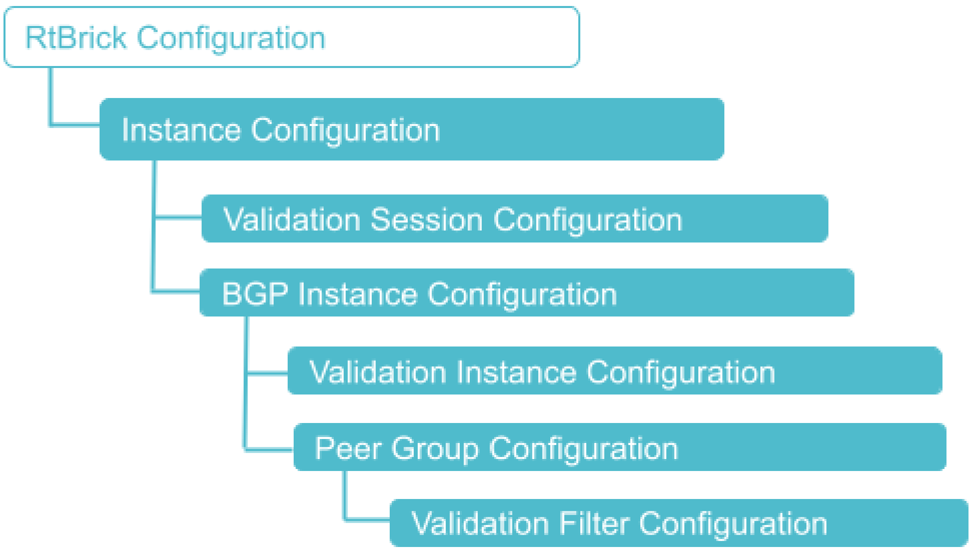
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

2.5.2. BGP Prefix Validation Using RPKI-RTR Configuration

Configuration Hierarchy

The diagram illustrates the BGP prefix validation mechanism configuration hierarchy.


RtBrick Configuration
Instance Configuration
Validation Session Configuration
BGP Instance Configuration
Validation Instance Configuration
Peer Group Configuration
Validation Filter Configuration

Configuration Syntax and Commands

The following sections describe the BGP RPKI prefix validation configuration syntax and commands.

Configure Validation Session with RPKI Local Cache

The following configuration commands are used to define validation session on an RBFS device. It specifies different session parameters such as cache IP address, update source, port, and preference.

In scenarios where multiple caches are available, the system employs a deterministic method for selecting the preferred cache. The cache preference method provides a way to determine which cache should be used based on configurable priorities.

Cache Preference:

1. When a preference is configured, the cache with the numerically lower preference value is selected.
2. If no preference is configured, or two caches have the same preference value, the selection follows these rules:

    Among IPv4 caches, the cache with the higher IP address is preferred.

    Among IPv6 caches, the cache with the higher IP address is preferred.

    Between IPv4 and IPv6 caches, the IPv4 cache takes precedence over IPv6.

Syntax

set instance <instance-name> **protocol validation session [IPv4|IPv6]** <cache-address> <update-source>

Attribute	Description
cache-address	Cache server IP address.
update-source	Source IP address.
port	Port number used fof the RPKI local cache. Range 1 - 65535
preference	Enables RBFS to select the validated route record from the preferred cache. Range 1 - 255. A numerically lower preference value indicates the higher preference.

Example: Validation session with the RPKI local cache

```
supervisor@rtbrick.net: cfg> show config instance vrf-blue protocol validation
{
  "rtbrick-config:validation": {
    "session": {
      "ipv4": [
        {
          "cache-address": "192.168.5.50",
          "update-source": "192.168.5.20",
          "port": 3323,
          "preference": 100
        }
      ],
      "ipv6": [
        {
          "cache-address": "192:168:5::50",
          "update-source": "192:168:5::20",
          "port": 3323,
          "preference": 10
        }
      ]
    }
  }
}
```

The preceding configuration sets up RPKI validation for both IPv4 and IPv6 routes in BGP. IPv4 session connects the router to an RPKI cache server with the RTR protocol, using the specified cache server IP address 192.168.5.50. The update-source, that is source IP address, is specified to 192.168.5.20 for the IPv4 session. The port number that is used for RTR communication is specified to '3323' for the session.

For the IPv6 session, the cache server IP address is specified to 192:168:5::50 and the source IP is specified 192:168:5::20. The port number is specified 3323 that is used for RTR communication.

The preference value for the IPv4 session is set to 100, while the preference value for the IPv6 session is set to 10. The preference value determines which session is prioritized when both sessions are available. A lower numerical preference value indicates a higher preference. In this scenario, the IPv6 session has a higher preference because it has a lower value, meaning that the IPv6 session will take priority over the IPv4 session.

Configure BGP Validation Instance

Syntax

set instance <instance-name> **protocol bgp validation-instance** <validation-instance>

Attribute	Description
<validation-instance>	Validation instance name.

Example: Validation Instance configuration

```
supervisor@rtbrick.net: cfg> show config instance default protocol bgp validation-
instance
{
  "rtbrick-config:validation-instance": "vrf-blue"
}
```

The configuration sets up 'vrf-blue' as the validation instance.

Configure BGP Validation Filter

Validation filter can be configured under a specific BGP peer group.

Syntax

set instance <instance-name> **protocol bgp peer-group** <pg-name> **validation-filter** [strict|loose|disable]

Attribute	Description
strict	BGP accepts only the valid routes.
loose	BGP accepts both valid routes and unknown routes. But it rejects routes with invalid RPKI data.
disable	No route validation is performed. Routes are installed based on the BGP route selection process.

Example: Validation filter configuration

```
supervisor@rtbrick.net: cfg> show config instance default protocol bgp peer-group
P2 validation-filter
{
  "rtbrick-config:validation-filter": "loose"
}
```

The configuration defines the validation filter. The validation filter is defined as 'loose' for the configuration that means BGP can accept both valid routes and unknown routes, but it rejects routes with invalid RPKI data.

2.5.3. BGP Route Validation with RPKI-RTR Operational Commands

BGP RPKI-RTR Show Commands

The BGP RPKI-RTR show commands provide detailed information about the BGP RPKI-RTR route validation operations.

Prefix Validation

The show validation command displays detailed validation information.

Syntax:

show validation <option>

Option	Description
cache-in	Displays prefix information per cache IP and source IP.
database	Displays information related to the validation database.
session	Provides details about validation sessions.

To access the Operational State API that corresponds to this CLI, click here.

Validation Session

The show validation session command and options provide information about the TCP session information between RBFS and the local caches.

Syntax:

show validation session <option>

Option	Description
-	Without any option, it displays information about all the validation session.
cache	Displays validation session information filtered by a specific cache address.

Option	Description
detail	Provides detailed information about each validation session.
instance	Displays validation session information for a specific instance.
source	Displays validation session information filtered by a specific source IP address.

Example: Validation session

```
supervisor@rtbrick.net: cfg> show validation session
Instance: vrf-blue
  Cache                         Source                    State         Up/Down
Time      IPv4 Rcvd   IPv6 Rcvd
  192.168.5.50                  192.168.5.20              Established
0d:00h:34m:12s     4048744      986282
  192:168:5::50                 192:168:5::20             Established
0d:00h:34m:12s     4048734      986261
```

Validation Cache-in

The show validation cache-in command provides cache information.

Syntax:

show validation cache-in <option>

The show command and options provide detailed information about the received validation route records from the local caches.

Option	Description
-	Without any option, it displays information about the prefix validation data for all caches and source IPs.
afi	Filters the cache-in validation data based on the address family, such as IPv4 or IPv6.
cache	Displays validation information related to a specific cache address.
instance	Filters the validation data for a specific instance.
prefix	Displays validation information for a specific prefix.

Example: Information for the specific cache.

```
supervisor@rtbrick.net: cfg> show validation cache-in instance default cache
10.1.1.2
Instance: default  AFI: ipv4
  Cache IP: 10.1.1.2  Source IP: 10.1.1.1  Total Prefixes: 443092
    Prefix                                     As Num    Max Len
    1.0.0.0/24                                  13335         24
    3.0.0.0/10                                  16509         24
    3.0.0.0/15                                  16509         24
    5.0.0.0/19                                  29256         19
    20.0.0.0/11                                  8075         11
    23.0.0.0/12                                 20940         12
    23.0.0.0/24                                 20940         24
    24.0.0.0/12                                  7922         12
    24.0.0.0/16                                 33659         16
    27.0.0.0/22                                 16509         24
    31.0.0.0/16                                  8374         16
    <...>
```

Validation Database

The validation database contains preferred validation route records based the cache preference. The show validation database provides detailed information about the validation database, which is used for prefix validation.

Syntax:

show validation database <option>

Option	Description
-	Without any option, it displays the information about all validation databases.
afi	Filters the validation database based on the address family, such as IPv4 or IPv6.
as-num	Displays validation information for a specific Autonomous System (AS).
cache	Displays database entries associated with a specific cache address.
instance	Displays database information for a specific instance.
prefix	Filters the database to show validation details for a specific prefix.
summary	Provides a summary of the validation database.

Example: Validation database information

```
supervisor@S1-STD-1-1012>bm14-tst.fsn.rtbrick.net: cfg> show validation database
Instance: vrf-blue, AFI: ipv4
  Prefix                                      As Num    Max Len    Cache
  1.0.0.0/24                                   13335         24    192:168:5::50
  3.0.0.0/10                                   16509         24    192:168:5::50
  3.0.0.0/15                                   16509         24    192:168:5::50
  5.0.0.0/19                                   29256         19    192:168:5::50
  20.0.0.0/11                                   8075         11    192:168:5::50
  23.0.0.0/12                                  20940         12    192:168:5::50
  23.0.0.0/24                                  20940         24    192:168:5::50
```

Example: Validation database information for a specific AS number.

```
supervisor@rtbrick.net: cfg> show validation database as-num 5089
Instance: vrf-blue, AFI: ipv4
  Prefix                                      As Num    Max Len    Cache
  80.0.0.0/13                                   5089         17    192:168:5::50
  82.0.0.0/12                                   5089         12    192:168:5::50
  82.0.0.0/13                                   5089         13    192:168:5::50
  82.0.0.0/14                                   5089         14    192:168:5::50
  86.0.0.0/11                                   5089         11    192:168:5::50
  86.0.0.0/13                                   5089         13    192:168:5::50
  86.0.0.0/14                                   5089         14    192:168:5::50
  86.0.0.0/15                                   5089         15    192:168:5::50
  86.0.0.0/16                                   5089         16    192:168:5::50
  86.1.0.0/16                                   5089         16    192:168:5::50
```

Example: Validation database summary

```
supervisor@rtbrick.net: cfg> show validation database summary
Instance: vrf-blue
IPv4 Prefix: 452282
IPv6 Prefix: 113149
```

Clear Commands

Clear Validation Session

The clear validation session command and options are used to remove or reset data related to validation session.

Syntax:

clear validation session <option>

Option	Description
all	Resets all existing validation sessions (without clearing the database).
all soft	Refreshes the only the validation database by sending a reset query to the cache.
instance	Resets validation sessions per instance.

Example Commands:

```
clear validation session all
clear validation session all soft
clear validation session instance vrf-blue
clear validation session instance vrf-blue 192.168.5.50 source 192.168.5.20 soft
```

To access the Operational State API that corresponds to this CLI, click here.

Clear Validation Database

The clear validation database command is used to remove entries in the validation database.

Syntax:

clear validation database <option>

Option	Description
all	Clears all entries in the validation database.
instance	Clears validation database for a specific instance.

Example Commands:

```
clear validation database all
clear validation database instance vrf-blue
clear validation database instance vrf-blue 192.168.5.50 source 192.168.5.20
```

To access the Operational State API that corresponds to this CLI, click

here.

2.6. IS-IS

2.6.1. IS-IS Overview

IS-IS, or Intermediate System to Intermediate System, is an open standard routing protocol. ISO published the standard as a way to route datagrams as part of their OSI stack. IETF later republished the standard, and added IP route support.

It is a link-state routing protocol, similar to OSPF. It forms neighbor adjacencies, has areas, exchanges link-state packets, builds a link-state database and runs the Dijkstra SPF algorithm to find the best path to each destination, which is installed in the routing table.

IS-IS in RBFS supports segment routing based on RFC 8667. IS-IS Segment Routing enhances the IS-IS protocol by introducing new TLVs to carry segment routing information within IS-IS protocol packets. It introduces Segment IDs (SIDs) to represent different types of segments, such as node SIDs and adjacency SIDs.

RFC and draft compliance are partial except as specified.

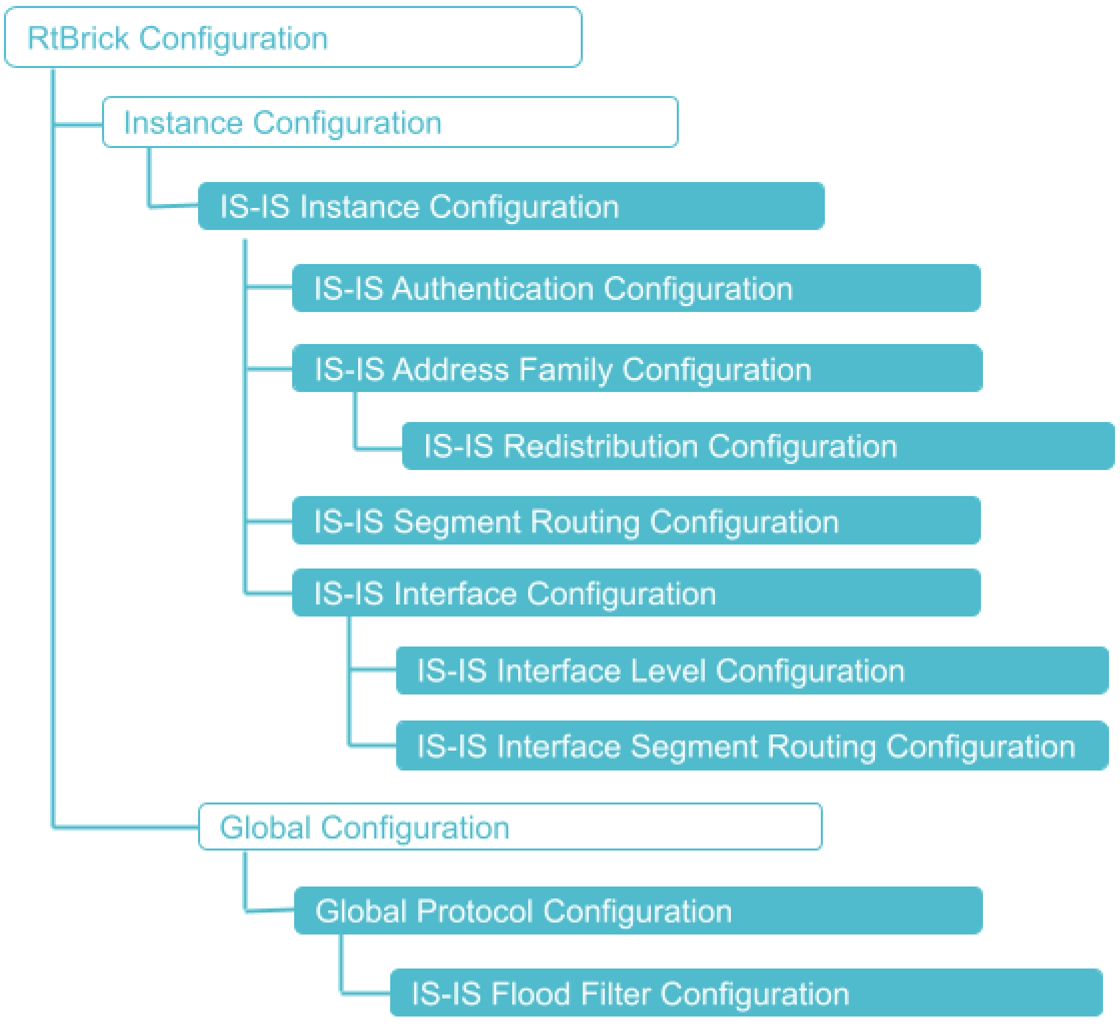
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

2.6.2. IS-IS Configuration

Configuration Hierarchy

The diagram below illustrates the IS-IS configuration hierarchy.


RtBrick Configuration
Instance Configuration
IS-IS Instance Configuration
IS-IS Authentication Configuration
IS-IS Address Family Configuration
IS-IS Redistribution Configuration
IS-IS Segment Routing Configuration
IS-IS Interface Configuration
IS-IS Interface Level Configuration
IS-IS Interface Segment Routing Configuration
Global Configuration
Global Protocol Configuration
IS-IS Flood Filter Configuration

Configuration Syntax and Commands

The following sections describe the IS-IS configuration syntax and commands.

Instance Configuration

The instance configuration hierarchy includes parameters that are required for or used by IS-IS.

Syntax:

set instance <instance-name> **protocol isis** <attribute> <value>

Attribute	Description
<name>	Name of the IS-IS instance

Attribute	Description
area <area>	IS-IS area-address. The area can be represented in 1, 3, 5, 13 bytes format.
authentication <…>	Specifies the authentication scheme for IS-IS. Refer to section IS-IS Authentication Configuration for the IS-IS authentication configuration details.
holding-time <holding-time>	Specifies how long a neighbor should consider this routing device to be operative without receiving another hello packet. Default value: 30 seconds Range: 3 to 180 seconds
hostname <hostname>	Specifies the hostname mapped to the system identifier.
ignore-attached-bit [true/false]	This configuration allows you to enable the routing device to ignore the attached bit on incoming Level 1 link-state PDUs. If the attached bit is ignored, no default route, which points to the routing device which has set the attached bit, is installed.
interface <…>	Name of the interface. Refer to section Configuring IS-IS Interface for the interface configuration details.
ipv6-disable [true/false]	Specifies whether the ipv6-disable configuration is enabled or not. When you set this value to "true", it indicates that IPv6 configuration is disabled.
[level-1/level-2] address-family <…>	Protocol ISIS level-1/level-2 address-family configuration. Refer to section IS-IS Address-Family Configuration for the address family configuration details.
level1-to-level2 route-leak [enable/disable]	Specifies whether the level1-to-level2 route-leak is enabled or not. When set to disable, IS-IS will not leak routing information from a Level 1 area to a Level 2 area. By default, this option is enabled.
lsp-lifetime <lsp-lifetime>	IS-IS link-state PDUs maximum lifetime. Default value: 65535 seconds

Attribute	Description
multipath <multipath>	Load sharing among multiple ISIS paths. Default value: 256
no-mpls-transit-path [true/false]	When set to true, IS-IS will not install segment routing transit path. Default: false
overload [true/false]	When set to true, IS-IS overload bit is set. Default: false
router-id <router-id>	ISIS router identifier (ipv4 format: A.B.C.D)
system-id <system-id>	Specifies the system ID of the device.

Example 1: IS-IS Instance Configuration

```
supervisor@rtbrick>spine1: cfg>show config instance default protocol isis
{
  "rtbrick-config:isis": {
    "system-id": "1921.6800.1001",
    "area": [
      "49.0001/24"
      ],
    "hostname": "isr1",
 <...>
```

Example 2: Disabling IS-IS Route Leaking from a Level 1 Area to a Level 2 Area

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis level1-
to-level2
{
  "rtbrick-config:level1-to-level2": {
    "route-leak": "disable"
  }
}
```

IS-IS Authentication Configuration

Syntax:

set instance <instance-name> **protocol isis authentication** [**level-1** | **level-2**] <attribute> <value>

Attribute	Description
check [disable / enable]	Specifies an authentication check to reject PDUs that do not match the type or key requirements. You can enable or disable the authentication check.
key-id1 <key-id1> / key-id2 <key-id2>	The key ID allows you to specify the key identifiers for level-1/level-2 authentication.
key1-encrypted-text <key1-encrypted-text> / key2-encrypted-text <key2-encrypted-text>	Authentication key1 and key 2 encrypted text
key1-plain-text <key1-plain-text> / key2-plain-text <key2-plain-text>	The level-1/level-2 authentication keys specify the authentication keys (passwords) that are used by the neighboring routing devices to verify the authenticity of packets sent from this interface. For the key to work, you also must include the authentication-type statement.
type	Enables you to specify the authentication scheme for IS-IS. If you enable authentication, you must specify a password by including the authentication-key statement. The following authentication types are supported: • clear_text • md5 • sha1

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
authentication
{
  "rtbrick-config:authentication": {
    "level-1": {
      "type": "md5",
      "key1-encrypted-text": "$24a6f10525a11077bec3b451be8855877"
    },
    "level-2": {
      "type": "md5",
      "key1-encrypted-text": "$2d787015bf84f3b58d5fd393a96c9639c"
    }
  }
}
```

IS-IS Address-Family Configuration

The address-family command allows you to enable the address families that IS-IS will route and configure settings that are specific to that address family.

Syntax:

set instance <instance-name> **protocol isis** [**level-1** | **level-2**] **address-family** <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast or labeled-unicast

Configuring Route Redistribution

Syntax:

set instance <instance-name> **protocol isis** [**level-1** | **level-2**] **address-family** <afi> <safi> **redistribute** <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast or labeled-unicast
redistribute <protocol>	Specifies the source from which the routes are to be redistributed from. The available options include arp-nd, bgp, bgp-local, bgp-local-origin, direct, igmp, ospf, l2tpv2, ldp, local, pim, ppp, rib, and static.
redistribute <protocol> <policy>	Specifies the name of the policy map. The redistribute attach point allows routes from other sources to be advertised by IS-IS. Policy can be applied only to the routes that are redistributed from other sources to IS-IS. The support for inter-level leaking through policy is unavailable.

Example: IS-IS address-family configuration

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis level-1
{
  "rtbrick-config:level-1": {
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "unicast",
        "redistribute": [
          {
            "source": "static",
            "policy": "filter-link-address"
          }
        ]
      },
      {
        "afi": "ipv6",
        "safi": "unicast",
        "redistribute": [
          {
            "source": "static",
            "policy": "filter-link6-address"
          }
        ]
      }
    ]
  }
}
```

Segment Routing Configuration

IS-IS segment routing extensions allow to advertise labels with prefixes.

RBFS currently supports the following IS-IS segment routing features:

- MPLS data plane
- IPv4 prefixes (TLV 135) and IPv6 prefixes (TLV 236)
- Prefix SID with node flag (Node SID) on loopback interface
- Anycast SID
- A single global SRGB block
- Adjacency SIDs

Syntax:

set instance <instance-name> **protocol isis segment-routing** <attribute> <value>

Attribute	Description
srgb base <srgb base>	Specifies the segment routing global block (SRGB) in source packet routing. SRGB is used for prefix SIDs. Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. It is recommended to assign label values outside of these reserved ranges to avoid conflicts.
srgb range <srgb range>	IS-IS system range of labels from the base label.
srlb base <srlb base>	Specifies the segment routing local block (SRLB) in source packet routing. SRLB is used for adjacency SIDs. Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. It is recommended to assign label values outside of these reserved ranges to avoid conflicts.
srlb range <srlb range>	IS-IS system range of labels from the base label.

Example: IS-IS Segment Routing Configuration

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
segment-routing
{
  "rtbrick-config:segment-routing": {
    "srlb": {
      "base": 2000,
      "range": 1000
    },
    "srgb": {
      "base": 1000,
      "range": 1000
    }
  }
}
```

Configuring IS-IS Interface

By default, there are no interfaces associated with IS-IS. You must configure at least one IS-IS interface for IS-IS adjacency formation.

Syntax:

set instance <instance> **protocol isis interface** <name> <attribute> <value>

Attribute	Description
<name>	Specifies the name of the IS-IS interface.
flood-filter <flood-filter>	Specifies the IS-IS flood filter name
hello-interval	Specifies IS-IS system interface hello interval. To configure hello-interval, the user can either configure holding-time/hello-interval under interface or hello-interval under level [broadcast interface only]. The order of priority, from highest to lowest, is as follows: * For Broadcast interface type: hello-interval under interface level 1/2 > hello-interval under interface > holding-time under instance * For Point-to-Point interface type: hello-interval under interface > holding-time under instance
level-1 / level-2	Specify IS-IS interface level configuration. Refer to section IS-IS Interface Level Configuration for the IS-IS interface level configuration details.
lsp-interval <lsp-interval>	IS-IS system interface LSP interval. Default value: 100
passive [true / false]	Enable interface in passive mode. Default: false
system-id <system-id>	Interface level system id
type [broadcast / point-to-point]	Specifies the type of the IS-IS system interface. Default: point-to-point. broadcast—Specifies a broadcast (or LAN) interface. point-to-point—Specifies a point-to-point interface.
ldp-synchronization [enable / disable]	Enable LDP IGP synchronization. Default: disable

Example 1: IS-IS Interface Configuration

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface
{
  "rtbrick-config:interface": [
    {
      "name": "ifl-0/0/2/0",
      "type": "point-to-point",
      "passive": "false",
      "level-1": {
        "snp-authentication": "disable",
        "hello-authentication": "disable"
      },
      "level-2": {}
    },
    {
      "name": "ifl-0/0/2/1",
      "flood-filter": "spine1_lsr1_flood_filter",
      "type": "broadcast",
      "lsp-interval": 200,
      "ldp-synchronization": "enable",
      "level-1": {
        "snp-authentication": "enable",
        "hello-authentication": "enable",
        "metric": 1000,
        "adjacency-disable": "false",
        "priority": 100,
        "hello-interval": 20
      },
      "level-2": {
        "priority": 100,
        "hello-interval": 20
      }
    },
    {
      "name": "lo-0/0/0/0",
      "passive": "true",
      "segment-routing": {
        "ipv4": {
          "index": 100
        },
        "ipv6": {
          "index": 102
        }
      },
      "level-1": {},
      "level-2": {}
    }
  ]
}
```

Example 2: IS-IS Interface Level Flood Filter Configuration

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface ifl-0/0/2/1 flood-filter
{
  "rtbrick-config:flood-filter": "spine1_lsr1_flood_filter"
}
```

Example 3: IS-IS Interface Configuration with enabled LDP synchronization

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface ifl-0/0/2/1 ldp-synchronization
{
  "rtbrick-config:ldp-synchronization": "enable"
}
```

Example 4: IS-IS Interface Configuration for a Broadcast Interface.

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface ifl-0/0/2/1 type
{
  "rtbrick-config:type": "broadcast"
}
```

IS-IS Interface Level Configuration

Syntax:

set instance <instance> **protocol isis interface** <name> [**level-1** | **level-2**] <attribute> <value>

Attribute	Description
adjacency-disable [true/false]	Specify the level-1/level-2 adjacency on an interface. Default: false.
hello-authentication [disable/enable]	Authentication on hello packets.
hello-interval	Specifies the length of time between the sending of IS-IS hello PDUs. Default: 10. The hello interval can be set for both broadcast and point-to-point interfaces that are configured for Levels 1 and 2.
priority	Specify the priority on a broadcast interface. Default: 64.
metric <metric>	Level-1/Level-2 metric on an interface. Default: 1000000.
snp-authentication [enable/disable]	Authentication on CSNP/PSNP packets.

Example 1: IS-IS Interface Level Configuration

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface ifl-0/0/2/1
{
  "rtbrick-config:interface": [
    {
      "name": "ifl-0/0/2/1",
      "flood-filter": "spine1_lsr1_flood_filter",
      "type": "broadcast",
      "lsp-interval": 200,
      "ldp-synchronization": "enable",
      "level-1": {
        "snp-authentication": "enable",
        "hello-authentication": "enable",
        "metric": 1000,
        "adjacency-disable": "false",
        "priority": 100,
        "hello-interval": 20
      },
      "level-2": {
        "priority": 100,
        "hello-interval": 20
      }
    }
  ]
}
```

Interface-level Segment Routing Configuration

Syntax:

set instance <instance> **protocol isis interface** <name> **segment-routing** <attribute> <value>

Attribute	Description
segment-routing [ipv4 / ipv6] anycast-index <anycast-index>	Anycast index segment-ID. The prefix SIDs and anycast SIDs are applied on loopback interface only.
segment-routing [ipv4 / ipv6] index <index>	Prefix index segment ID.
segment-routing point-to-point [ipv4 / ipv6] adjacency-index <adjacency-index>	Adjacency index segment-ID. The adjacency SIDs are applied on active IS-IS interfaces on which adjacencies are established.

Example 1: IS-IS Interface Level Segment Routing Configuration for Prefix and Anycast SID

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface
{
  "rtbrick-config:instance": [
      {
        "name": "default",
        "protocol": {
          "isis": {
            "interface": [
              {
                "name": "lo-0/0/0",
                "segment-routing": {
                  "ipv4": {
                    "index": 100
                  },
                  "ipv6": {
                    "index": 200
                  }
                }
              },
              {
                "name": "lo-0/0/1",
                "segment-routing": {
                  "ipv4": {
                    "anycast-index": 110
                  },
                  "ipv6": {
                    "anycast-index": 210
                  }
                }
              }
            ]
          }
        }
      }
    ]
  }
}
```

Example 2: IS-IS Interface Level Segment Routing Configuration for Adjacency SID

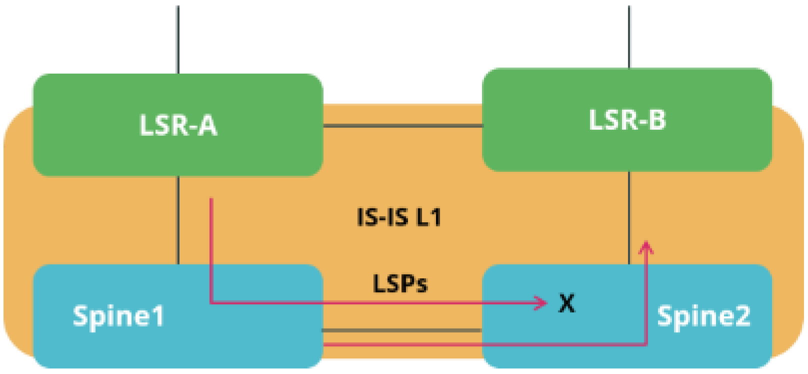
```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface ifl-0/0/2/0 segment-routing
{
  "rtbrick-config:segment-routing": {
    "point-to-point": {
      "ipv4": {
        "adjacency-index": 111
      },
      "ipv6": {
        "adjacency-index": 112
      }
    }
  }
}
```

IS-IS Global Configuration

IS-IS Flood Filter Configuration

In IS-IS, by default all routers flood link-state packets, so that all routers will have a complete topology view. IS-IS flood filters allow to modify this behavior and limit the exchange of LSPs. For example, if two spine routers in a spine/leaf fabric are symmetrically connected to two upstream label-switch routers (LSR) like shown in the figure below, you can use a flood filter to not advertise LSPs learned from LSR A back to the LSR B via the second spine switch.

The flooding filter configuration is part of the global configuration hierarchy and therefore you can configure filtering globally, i.e. not per instance, so that the filter configurations can be reused across instances.



Syntax:

set global protocol isis flood-filter <filter-name> <ordinal> <attribute> <value>

Attribute	Description
<filter-name>	Filter-name which binds a flooding filter to an IS-IS interface
<ordinal>	Number to filter rule
action [block/flood]	Action required to flood or not
ordinal-name <ordinal-name>	Name for the filter rule
system-id <system-id>	IS-IS instance system-id
system-id-mask <system-id-mask>	System ID mask on which the filter should match

Example: IS-IS Flood Filter Configuration

```
supervisor@rtbrick>spine1: cfg> show config instance default protocol isis
interface ifl-0/0/2/1 flood-filter
{
  "rtbrick-config:flood-filter": "spine1_lsr1_flood_filter"
}
```

2.6.3. IS-IS Operational Commands

IS-IS Show Commands

The IS-IS show commands provide detailed information about the IS-IS protocol operation and IS-IS routes.

IS-IS Overview

Syntax:

show isis overview

Option	Description
-	Without any option, this command displays a summary of all the IS-IS instances

Example: Summary view of all the IS-IS instances

```
supervisor@rtbrick>spine1: op> show isis overview
Instance: default
  System ID: 1921.6800.1001
  System hostname: isr1
  Areas: 49.0001/24
  Neighbor hold time: 30 sec
  LSP life time: 65535 sec
  Overload bit set: False
  SRGB base: 1000
  SRGB range: 1000
  SRGB label values: 1000 - 1999
  SRLB base: 2000
  SRLB range: 1000
  SRLB label values: 2000 - 2999
  Authentication: Level 1: md5, Level 2: md5
```

To access the Operational State API that corresponds to this CLI, click

here.

IS-IS Interface

Syntax:

show isis interface <option>

Option	Description
-	Without any option, this command displays a summary of all the IS-IS interfaces
instance	Displays IS-IS interface information for an instance
statistics	Displays IS-IS interface statistics information
detail	Displays detailed output for all interfaces.

Example 1: Summary view of the IS-IS interfaces

```
supervisor@rtbrick>spine1: op> show isis interface
Instance: default
  Interface      Level   Adjacencies    Metric     Type            Passive  LDP-
sync
  lo-0/0/0/0         1             0    1000000    loopback              1  -
  lo-0/0/0/0         2             0    1000000    loopback              1  -
  ifl-0/0/2/0        2             1    1000000    point-to-point        0  -
  ifl-0/0/2/1        1             2    1000000    broadcast             0  -
  ifl-0/0/2/1        2             2    1000000    broadcast             0  -
```

Example 2: Summary view of the IS-IS interfaces for a specific instance

```
supervisor@rtbrick>spine1: op> show isis interface instance default
Instance: default
  Interface      Level   Adjacencies  Metric     Type            Passive  LDP-sync
  lo-0/0/0/0         1             0  1000000    loopback              1  -
  lo-0/0/0/0         2             0  1000000    loopback              1  -
  ifl-0/0/2/0        2             1  1000000    point-to-point        0  -
  ifl-0/0/2/1        1             2  1000000    broadcast             0  -
  ifl-0/0/2/1        2             2  1000000    broadcast             0  -
```

Example 3: Summary view of the IS-IS interfaces for a specific interface

```
supervisor@rtbrick>spine1: op> show isis interface ifl-0/0/2/0
Instance: default
  Interface: ifl-0/0/2/0, Level: 1
    Type: point-to-point, Passive: False
    Metric: 1000000
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Adjacencies: 1
    CNSP in: 0, CNSP out: 2446, CNSP success: 1222, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 0, LSP out: 0, LSP success: 0, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 1134, IIH out: 1138
  Interface: ifl-0/0/2/0, Level: 2
    Type: point-to-point, Passive: False
    Metric: 1000000
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Adjacencies: 1
    CNSP in: 0, CNSP out: 2446, CNSP success: 1222, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 0, LSP out: 0, LSP success: 0, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 1134, IIH out: 1138
```

Example 4: Summary view of the IS-IS interface statistics

```
supervisor@rtbrick>spine1: op> show isis interface statistics
Instance: default
  Interface        Level  CSNP In  CSNP Out  CSNP Fail  PSNP In  PSNP Out  PSNP
Fail  LSP In  LSP Out  LSP Fail  IIH In
 IIH Out
  lo-0/0/0/0          1        0         0          0        0         0
0       0        0         0       0
       0
  lo-0/0/0/0          2        0         0          0        0         0
0       0        0         0       0        0
  ifl-0/0/2/0         1        0      2454          0        0         0
0       0        0         0    1138
    1142
  ifl-0/0/2/0         2        0      2454          0        0         0
0       0        0         0    1138
    1142
  ifl-0/0/2/1       1        0         0          0        0         0          0
3        1         0    2288
    2285
  ifl-0/0/2/1         2        0         0          0        0         0
0       4        1         0    2288     2285
```

Example 5: Detailed output of the IS-IS interface

```
supervisor@rtbrick>spine1: op> show isis interface detail
Instance: default
  Interface: lo-0/0/0/0, Level: 1
    Type: loopback, Passive: True
    Metric: 1000000
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Adjacencies: 0
    CNSP in: 0, CNSP out: 0, CNSP success: 0, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 0, LSP out: 0, LSP success: 0, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 0, IIH out: 0
  Interface: lo-0/0/0/0, Level: 2
    Type: loopback, Passive: True
    Metric: 1000000
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Adjacencies: 0
    CNSP in: 0, CNSP out: 0, CNSP success: 0, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 0, LSP out: 0, LSP success: 0, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 0, IIH out: 0
Instance: default
  Interface: ifl-0/0/2/0, Level: 1
    Type: point-to-point, Passive: False
    Metric: 1000000
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Adjacencies: 1
    CNSP in: 0, CNSP out: 2460, CNSP success: 1229, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 0, LSP out: 0, LSP success: 0, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 1141, IIH out: 1145
  Interface: ifl-0/0/2/0, Level: 2
    Type: point-to-point, Passive: False
    Metric: 1000000
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Adjacencies: 1
    CNSP in: 0, CNSP out: 2460, CNSP success: 1229, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 0, LSP out: 0, LSP success: 0, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 1141, IIH out: 1145
 <...>
```

Example 5: Detailed output of an IS-IS interface that supports broadcast

```
supervisor@rtbrick>spine1: op> show isis interface ifl-0/0/2/1
Instance: default
  Interface: ifl-0/0/2/1, Level: 1
    Type: broadcast, Passive: False, Circuit ID: 0x1
    Metric: 1000000, Priority: 64
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Designated IS: isr6.01(1921.6800.1006.01)
    Adjacencies: 2
    CNSP in: 0, CNSP out: 0, CNSP success: 1238, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 3, LSP out: 1, LSP success: 3, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 2304, IIH out: 2301
  Interface: ifl-0/0/2/1, Level: 2
    Type: broadcast, Passive: False, Circuit ID: 0x1
    Metric: 1000000, Priority: 64
    Hello interval: 10 seconds, Hello multiplier: 3, Hold time: 30 seconds
    Designated IS: isr6.01(1921.6800.1006.01)
    Adjacencies: 2
    CNSP in: 0, CNSP out: 0, CNSP success: 1238, CNSP fail: 0
    PSNP in: 0, PSNP out: 0, PSNP success: 0, PSNP fail: 0
    LSP in: 4, LSP out: 1, LSP success: 4, LSP fail: 0, LSP in purge: 0 LSP in
auth fail: 0
    IIH in: 2304, IIH out: 2301
```

```
supervisor@rtbrick>spine1: op>
```

To access the Operational State API that corresponds to this CLI, click here. here.

IS-IS Neighbor

Syntax:

show isis neighbor <option>

Option	**Description**
-	Without any option, this command displays a summary of all the IS-IS neighbors
detail	Displays detailed information for IS-IS neighbor
instance	Displays IS-IS neighbor information for an instance

Example 1: Summary view of the IS-IS neighbor

```
supervisor@rtbrick>spine1: op> show isis neighbor
Instance: default
  Interface        System       Level   State   Type       Up since
Expires
  ifl-0/0/2/0        isr6         L1L2    Up      P2P        Thu Jul 25 05:40:05
in 18s 750197us
  ifl-0/0/2/1        isr6         L1      Up      Broadcast  Thu Jul 25 05:39:58
in 26s 728457us
  ifl-0/0/2/1        isr6         L2      Up      Broadcast  Thu Jul 25 05:39:58
in 29s 729531us
```

Example 2: Detailed view of the IS-IS neighbor

```
supervisor@S1-STD-17-1703: cfg> show isis neighbor detail
Instance: default
  System: isr6, Interface: ifl-0/0/2/0
    State: Up, Level: L1L2, Adjacency type: P2P
    Hold time: 30.0s, Expiry time: in 29s 780091us
    Local IPv4 address: 16.0.1.1, Remote IPv4 address: 16.0.1.6
    Local IPv6 address: fe80::7807:2dff:fec0:2, Remote IPv6 address:
fe80::7847:eff:fec0:0
    Up since: Thu Jul 25 05:40:05 GMT +0000 2024, Last down reason: NA
    Last transition: 2024-07-25T05:40:05.778348+0000, Number of transitions: 2
    Error counters:
      Level mismatch: 0, Area mismatch: 0, System ID: 0, Subnet mismatch: 0
      Hold timeout: 0, Neighbor down: 0, Interface down: 0, Admin reset: 0
      Interface configuration: 0, Area configuration: 0, Other: 0
  System: isr6, Interface: ifl-0/0/2/1
    State: Up, Level: L1L2, Adjacency type: Broadcast
    Hold time: 30.0s, Expiry time: in 24s 779280us
    Priority: 64, Designated IS: 1921.6800.1006.01, SNPA: 7a:47:0e:c0:00:00
    Local IPv4 address: 16.0.2.1, Remote IPv4 address: 16.0.2.6
    Local IPv6 address: fe80::7807:2d00:1c0:2, Remote IPv6 address:
fe80::7847:e00:1c0:0
    Up since: Thu Jul 25 05:39:58 GMT +0000 2024, Last down reason: NA
    Last transition: 2024-07-25T05:39:58.725007+0000, Number of transitions: 3
    Error counters:
      Level mismatch: 0, Area mismatch: 0, System ID: 0, Subnet mismatch: 0
      Hold timeout: 0, Neighbor down: 0, Interface down: 0, Admin reset: 0
      Interface configuration: 0, Area configuration: 0, Other: 0
<...>
```

Example 3: Summary view of the IS-IS neighbor for the specified instance

```
supervisor@rtbrick>spine1: op> show isis neighbor instance default
Instance: default
  Interface      System      Level   State   Type       Up since
Expires
  ifl-0/0/2/0    isr6        L1L2    Up      P2P        Thu Jul 25 05:40:05    in
21s 893028us
  ifl-0/0/2/1    isr6        L1      Up      Broadcast  Thu Jul 25 05:39:58    in
26s 848320us
  ifl-0/0/2/1    isr6        L2      Up      Broadcast  Thu Jul 25 05:39:58    in
24s 845934us
supervisor@rtbrick>spine1: op>
```

Example 4: Detailed view of the IS-IS neighbor for the specified instance

```
supervisor@rtbrick>spine1: op> show isis neighbor instance default detail
Instance: default
  System: isr6, Interface: ifl-0/0/2/0
    State: Up, Level: L1L2, Adjacency type: P2P
    Hold time: 30.0s, Expiry time: in 21s 155501us
    Local IPv4 address: 16.0.1.1, Remote IPv4 address: 16.0.1.6
    Local IPv6 address: fe80::7807:2dff:fec0:2, Remote IPv6 address:
fe80::7847:eff:fec0:0
    Up since: Thu Jul 25 05:40:05 GMT +0000 2024, Last down reason: NA
    Last transition: 2024-07-25T05:40:05.778348+0000, Number of transitions: 2
    Error counters:
      Level mismatch: 0, Area mismatch: 0, System ID: 0, Subnet mismatch: 0
      Hold timeout: 0, Neighbor down: 0, Interface down: 0, Admin reset: 0
      Interface configuration: 0, Area configuration: 0, Other: 0
  System: isr6, Interface: ifl-0/0/2/1
    State: Up, Level: L1L2, Adjacency type: Broadcast
    Hold time: 30.0s, Expiry time: in 23s 150249us
    Priority: 64, Designated IS: 1921.6800.1006.01, SNPA: 7a:47:0e:c0:00:00
    Local IPv4 address: 16.0.2.1, Remote IPv4 address: 16.0.2.6
    Local IPv6 address: fe80::7807:2d00:1c0:2, Remote IPv6 address:
fe80::7847:e00:1c0:0
    Up since: Thu Jul 25 05:39:58 GMT +0000 2024, Last down reason: NA
    Last transition: 2024-07-25T05:39:58.725007+0000, Number of transitions: 3
    Error counters:
      Level mismatch: 0, Area mismatch: 0, System ID: 0, Subnet mismatch: 0
      Hold timeout: 0, Neighbor down: 0, Interface down: 0, Admin reset: 0
      Interface configuration: 0, Area configuration: 0, Other: 0
 <...>
```

To access the Operational State API that corresponds to this CLI, click here.

Designated IS (DIS) Election History

In IS-IS broadcast networks, the Designated Intermediate System (DIS) is elected to optimize the distribution of routing information, reducing network overhead by coordinating the flooding of Link State Packets (LSPs) among multiple routers. This ensures efficient and stable network operation. The following command displays the DIS election history in RBFS.

Syntax:

show isis dis-election history <attribute> <value>

Option	Description
-	Without any option, this command displays a summary of all the IS-IS.
instance	Displays DIS history information for the specified instance.
interface <interface-name>	Displays IS-IS DIS history information for the specified interface.
level [level-1/level-2]	Displays IS-IS DIS history information for the specified level.

Example: Summary view of IS-IS DIS Election History

```
supervisor@rtbrick>spine1: op> show isis dis-election history
Instance: default, Interface: ifl-0/0/2/1, Level: 1
Timestamp                             DIS                      Event
Thu Jul 25 05:39:58 GMT +0000 2024     -                       Neighbour
1921.6800.1006 adjacency state changed from Init to Up
Thu Jul 25 05:39:58 GMT +0000 2024    1921.6800.1006.01        Neighbour
1921.6800.1006 DIS changed from 0000.0000.0000.00 to 1921.6800.1006.01
Instance: default, Interface: ifl-0/0/2/1, Level: 2
Timestamp                             DIS                      Event
Thu Jul 25 05:39:58 GMT +0000 2024     -                       Neighbour
```

```
1921.6800.1006 adjacency state changed from Init to Up
Thu Jul 25 05:39:58 GMT +0000 2024    1921.6800.1006.01        Neighbour
1921.6800.1006 DIS changed from 0000.0000.0000.00 to 1921.6800.1006.01
```

IS-IS Hostname

Syntax:

show isis hostname

Option	Description
-	Without any option, this command displays a summary of all the IS-IS dynamic hostnames

Example: Summary view of IS-IS hostnames

```
supervisor@rtbrick>spine1: op> show isis hostname
Instance       System-ID           Hostname
default        1921.6800.1001      isr1
default        1921.6800.1006      isr6
default        1921.6800.1006      isr6
supervisor@rtbrick>spine1: op>
```

IS-IS Database

Syntax:

show isis database <option>

Option	Description
-	Without any option, this command displays all the IS-IS databases
detail	Displays detailed information for IS-IS database
instance	Displays IS-IS database information for an instance
lsp <lsp-id>	Displays a summary of IS-IS database for the specified LSP ID. This command includes an option for entering the system ID part either by hostname or by ID.
[level-1/level-2] lsp	Displays a summary of IS-IS database LSP information for specified level

Option	Description
system <system-id>	Displays a summary of IS-IS database for all LSPs from a system
[level-1/level-2] system	Displays a summary of IS-IS database for all LSPs from a system on the specified level.
[level-1/level-2] detail	Displays detailed information for the specified level

Example 1: Summary view of the IS-IS Database

```
supervisor@S1-STD-7-7001>bm01-tst.fsn.rtbrick.net: op> show isis database
Instance: default, Level: 1
  LSP ID                     Sequence     Checksum     Lifetime     Overload     Attached
  isr1.00-00                      0x4       0xe035        52785            0            0
  isr6.00-00                      0x3       0xe281        52785            0            0
  isr6.01-00                      0x1       0x4b1a        52778            0            0
Instance: default, Level: 2
  LSP ID                     Sequence     Checksum     Lifetime     Overload     Attached
  isr1.00-00                      0x6       0x75af        52785            0            0
  isr6.00-00                      0x5       0x117e        52785            0            0
  isr6.01-00                      0x1       0x4b1a        52778            0            0
```

Example 2: Summary view of the IS-IS database for the specified LSP

```
supervisor@rtbrick>spine1: op> show isis database lsp 1921.6800.1001.00-00
Instance: default, Level: 1
  LSP ID: isr1.00-00
    Interface:
    LSP Header:
      Sequence: 0x4
      Checksum: 0xe035
      Remaining lifetime: 65535 seconds
      Flags: Attached: 0, Overload: 0
    Packet:
      Length: 182 bytes
      Last received time: 2024-07-25T05:40:05.989755+0000
      Expiry: expires in 14h 39m 11s 284549us
    System ID: 1921.6800.1001
    Dynamic Hostname TLV (137): isr1
    Protocols Supported TLVs (129):
      Network layer protocol ID: IPv6
      Network layer protocol ID: IPv4
    Area Address TLVs (1):
      Area address: 49.0001
    IS Reachability TLVs (22):
      IS neighbor: 1921.6800.1006.00              Metric:  1000000
      IS neighbor: 1921.6800.1006.01              Metric:  1000000
    IPv4 Reachability TLVs (135):
      IPv4 prefix: 16.0.1.0/24                    Metric:  1000000   Internal   Up
      IPv4 prefix: 16.0.2.0/24                    Metric:  1000000   Internal   Up
      IPv4 prefix: 192.168.1.1/32                 Metric:  1000000   Internal   Up
SID:  100   Flags: Node
    IPv6 Reachability TLVs (236):
      IPv6 prefix: 16:0:1::/64                    Metric:  1000000   Internal   Up
      IPv6 prefix: 16:0:2::/64                    Metric:  1000000   Internal   Up
      IPv6 prefix: 192:168:1::1/128               Metric:  1000000   Internal   Up
SID:  102   Flags: Node
    Segment Routing TLVs (242/sub 2):
      SRGB: Base: 1000, Range: 1000, Router ID: 192.168.1.1
 <...>
```

Example 3: Detailed view of the IS-IS database for level-1

```
supervisor@rtbrick>spine1: op> show isis database level-1 detail
Instance: default, Level: 1
  LSP ID: isr1.00-00
    Interface:
    LSP Header:
      Sequence: 0x4
      Checksum: 0xe035
      Remaining lifetime: 52682 seconds
      Flags: Attached: 0, Overload: 0
    Packet:
      Length: 182 bytes
      Last received time: 2024-07-25T05:40:05.989755+0000
      Expiry: expires in 14h 38m 2s 951316us
    System ID: 1921.6800.1001
    Dynamic Hostname TLV (137): isr1
    Protocols Supported TLVs (129):
      Network layer protocol ID: IPv6
      Network layer protocol ID: IPv4
    Area Address TLVs (1):
      Area address: 49.0001
    IS Reachability TLVs (22):
      IS neighbor: 1921.6800.1006.00              Metric:  1000000,
      IS neighbor: 1921.6800.1006.01              Metric:  1000000,
    IPv4 Reachability TLVs (135):
      IPv4 prefix: 16.0.1.0/24                    Metric:  1000000   Internal   Up
      IPv4 prefix: 16.0.2.0/24                    Metric:  1000000   Internal   Up
      IPv4 prefix: 192.168.1.1/32                 Metric:  1000000   Internal   Up
SID:  100   Flags: Node
    IPv6 Reachability TLVs (236):
      IPv6 prefix: 16:0:1::/64                    Metric:  1000000   Internal   Up
      IPv6 prefix: 16:0:2::/64                    Metric:  1000000   Internal   Up
      IPv6 prefix: 192:168:1::1/128               Metric:  1000000   Internal   Up
SID:  102   Flags: Node
    Segment Routing TLVs (242/sub 2):
      SRGB: Base: 1000, Range: 1000, Router ID: 192.168.1.1
  LSP ID: isr6.00-00
    Interface: ifl-0/0/2/1
    LSP Header:
      Sequence: 0x3
      Checksum: 0xe281
      Remaining lifetime: 52682 seconds
      Flags: Attached: 0, Overload: 0
    Packet:
      Length: 182 bytes
      Last received time: 2024-07-25T05:40:05.786957+0000
      Expiry: expires in 14h 38m 2s 748444us
    System ID: 1921.6800.1006
    Dynamic Hostname TLV (137): isr6
    Protocols Supported TLVs (129):
      Network layer protocol ID: IPv6
      Network layer protocol ID: IPv4
    Area Address TLVs (1):
      Area address: 49.0001
    IS Reachability TLVs (22):
      IS neighbor: 1921.6800.1001.00              Metric:  1000000,
      IS neighbor: 1921.6800.1006.01              Metric:  1000000,
    IPv4 Reachability TLVs (135):
      IPv4 prefix: 16.0.1.0/24                    Metric:  1000000   Internal   Up
      IPv4 prefix: 16.0.2.0/24                    Metric:  1000000   Internal   Up
      IPv4 prefix: 192.168.1.6/32                 Metric:  1000000   Internal   Up
SID:  600   Flags: Node
    IPv6 Reachability TLVs (236):
      IPv6 prefix: 16:0:1::/64                    Metric:  1000000   Internal   Up
      IPv6 prefix: 16:0:2::/64                    Metric:  1000000   Internal   Up
      IPv6 prefix: 192:168:1::6/128               Metric:  1000000   Internal   Up
SID:  601   Flags: Node
    Segment Routing TLVs (242/sub 2):
      SRGB: Base: 6000, Range: 1000, Router ID: 192.168.1.6
 <...>
```

Example 4: Summary view of the IS-IS database for the specified instance

```
supervisor@rtbrick>spine1: op> show isis database instance default
Instance: default, Level: 1
  LSP ID                     Sequence     Checksum     Lifetime     Overload     Attached
  isr1.00-00                      0x4       0xe035        52625            0            0
  isr6.00-00                      0x3       0xe281        52625            0            0
  isr6.01-00                      0x1       0x4b1a        52618            0            0
Instance: default, Level: 2
  LSP ID                     Sequence     Checksum     Lifetime     Overload     Attached
  isr1.00-00                      0x6       0x75af        52625            0            0
  isr6.00-00                      0x5       0x117e        52625            0            0
  isr6.01-00                      0x1       0x4b1a        52618            0            0
```

Example 5: Summary view of the IS-IS database for the specified instance and LSP

```
supervisor@rtbrick>spine1: op> show isis database instance default lsp
1921.6800.1001.00-00
Instance: default, Level: 1
  LSP ID: isr1.00-00
    Interface:
    LSP Header:
      Sequence: 0x4
      Checksum: 0xe035
      Remaining lifetime: 65535 seconds
      Flags: Attached: 0, Overload: 0
    Packet:
      Length: 182 bytes
      Last received time: 2024-07-25T05:40:05.989755+0000
      Expiry: expires in 14h 36m 35s 357028us
    System ID: 1921.6800.1001
    Dynamic Hostname TLV (137): isr1
    Protocols Supported TLVs (129):
      Network layer protocol ID: IPv6
      Network layer protocol ID: IPv4
    Area Address TLVs (1):
      Area address: 49.0001
    IS Reachability TLVs (22):
      IS neighbor: 1921.6800.1006.00            Metric:  1000000
      IS neighbor: 1921.6800.1006.01            Metric:  1000000
    IPv4 Reachability TLVs (135):
      IPv4 prefix: 16.0.1.0/24                  Metric:  1000000   Internal   Up
      IPv4 prefix: 16.0.2.0/24                  Metric:  1000000   Internal   Up
      IPv4 prefix: 192.168.1.1/32               Metric:  1000000   Internal   Up
SID:  100   Flags: Node
    IPv6 Reachability TLVs (236):
      IPv6 prefix: 16:0:1::/64                  Metric:  1000000   Internal   Up
      IPv6 prefix: 16:0:2::/64                  Metric:  1000000   Internal   Up
      IPv6 prefix: 192:168:1::1/128             Metric:  1000000   Internal   Up
SID:  102   Flags: Node
    Segment Routing TLVs (242/sub 2):
      SRGB: Base: 1000, Range: 1000, Router ID: 192.168.1.1
 <...>
```

IS-IS Route

Syntax:

show isis route <option>

Option	Description
-	Without any option, this command displays a summary of the IS-IS route information
instance	Displays IS-IS route information for an instance
<afi> <safi>	Routing information for the specified AFI/SAFI. Supported SAFI values are 'unicast' and 'labeled-unicast'.

Example 1: Summary view of the IS-IS routes

```
supervisor@rtbrick>spine1: op> show isis route
Instance: default, AFI: ipv4, SAFI: unicast
  Prefix                    Level   Metric     Type         Next Hop       Interface
  16.0.1.0/24                   1   1000000    Internal     n/a            local
  16.0.2.0/24                   1   1000000    Internal     n/a            local
  192.168.1.1/32                1   1000000    Internal     n/a            local
  192.168.1.6/32                1   2000000    Internal     16.0.2.6       ifl-0/0/2/1
                                                            16.0.1.6       ifl-0/0/2/0
Instance: default, AFI: ipv4, SAFI: labeled-unicast
  Prefix                    Level   Metric     SID Index    Next Hop       Interface   Label
  192.168.1.6/32                1   2000000    600          16.0.2.6       ifl-0/0/2/1
                                                            16.0.1.6       ifl-0/0/2/0
Instance: default, AFI: ipv6, SAFI: unicast
  Prefix                    Level   Metric     Type         Next Hop       Interface
  16:0:1::/64                   1   1000000    Internal     n/a            local
  16:0:2::/64                   1   1000000    Internal     n/a            local
  192:168:1::1/128              1   1000000    Internal     n/a            local
  192:168:1::6/128              1   2000000    Internal     fe80::7847:eff:fec0:0     ifl-0/0/2/0
```

```
<...>
```

Example 2: Summary view of the IS-IS routes for the specified instance

```
supervisor@rtbrick>spine1: op> show isis route instance default
Instance: default, AFI: ipv4, SAFI: unicast
  Prefix                     Level   Metric     Type         Next Hop             Interface
  16.0.1.0/24                    1   1000000    Internal     n/a                  local
  16.0.2.0/24                    1   1000000    Internal     n/a                  local
  192.168.1.1/32                 1   1000000    Internal     n/a                  local
  192.168.1.6/32                 1   2000000    Internal     16.0.2.6             ifl-0/0/2/1
                                                             16.0.1.6             ifl-0/0/2/0
Instance: default, AFI: ipv4, SAFI: labeled-unicast
  Prefix                     Level   Metric     SID Index    Next Hop             Interface   Label
  192.168.1.6/32                 1   2000000    600          16.0.2.6             ifl-0/0/2/1
                                                             16.0.1.6             ifl-0/0/2/0
Instance: default, AFI: ipv6, SAFI: unicast
  Prefix                     Level   Metric     Type         Next Hop              Interface
  16:0:1::/64                    1   1000000    Internal     n/a                   local
  16:0:2::/64                    1   1000000    Internal     n/a                   local
  192:168:1::1/128               1   1000000    Internal     n/a                   local
  192:168:1::6/128               1   2000000    Internal     fe80::7847:eff:fec0:0 ifl-0/0/2/0
                                                             fe80::7847:e00:1c0:0  ifl-0/0/2/1
<...>
```

Example 3: Summary view of the IS-IS routes for the specified instance and address family (IPv4 unicast).

```
supervisor@rtbrick>spine1: op> show isis route instance default ipv4 unicast
Instance: default, AFI: ipv4, SAFI: unicast
  Prefix              Level   Metric     Type        Next Hop   Interface
  16.0.1.0/24             1   1000000    Internal    n/a        local
  16.0.2.0/24             1   1000000    Internal    n/a        local
  192.168.1.1/32          1   1000000    Internal    n/a        local
  192.168.1.6/32              2000000    Internal    16.0.2.6   ifl-0/0/2/1
                                                     16.0.1.6   ifl-0/0/2/0
```

Example 4: Summary view of the IS-IS routes for the specified instance and address family (IPv4 labeled-unicast).

```
supervisor@rtbrick>spine1: op> show isis route instance default ipv4 labeled-unicast
Instance: default, AFI: ipv4, SAFI: labeled-unicast
  Prefix         Level   Metric     SID Index    Next Hop        Interface   Label
  192.168.1.6/32    1    2000000    600          16.0.2.6        ifl-0/0/2/1
                                                 16.0.1.6        ifl-0/0/2/0
```

Example 5: Summary view of the IS-IS routes for the specified instance and address family (IPv6 unicast).

```
supervisor@rtbrick>spine1: op> show isis route instance default ipv6 unicast
Instance: default, AFI: ipv6, SAFI: unicast
```

```
  Prefix             Level   Metric     Type         Next Hop
Interface
  16:0:1::/64            1   1000000    Internal     n/a                     local
  16:0:2::/64            1   1000000    Internal     n/a                     local
  192:168:1::1/128       1   1000000    Internal     n/a                     local
  192:168:1::6/128       1   2000000    Internal     fe80::7847:eff:fec0:0   ifl-
0/0/2/0
                                                     fe80::7847:e00:1c0:0    ifl-
0/0/2/1
```

IS-IS Segment Routing

Syntax:

show isis segment-routing <option>

Option	Description
global-block	Displays Segment routing global block (SRGB) information
global-block instance <instance>	Displays Segment routing global block (SRGB) information for the specified instance
label-binding	Displays the IS-IS segment routing label bindings information
label-binding instance <instance>	Displays the IS-IS segment routing label bindings for the specified instance
prefix-segment	Displays the IS-IS prefix segments information
prefix-segment <instance>	Displays the IS-IS prefix segments for the specified instance
adjacency-segment	Displays the IS-IS segment routing adjacency SIDs

Example 1: Summary view of the IS-IS segment routing global block

```
supervisor@rtbrick>spine1: op> show isis segment-routing global-block
Instance: default, Level: 1
  System                    SRGB Base   SRGB Range
  isr1                           1000         1000
  isr6                           6000         1000
Instance: default, Level: 2
  System                    SRGB Base   SRGB Range
  isr1                           1000         1000
  isr6                           6000         1000
```

Example 2: Summary view of the IS-IS segment routing label binding

```
supervisor@rtbrick>spine1: op> show isis segment-routing label-binding
Instance: default, Level: 1
  System        Prefix                  Range   SID     Flags
Instance: default, Level: 2
  System        refix                   Range   SID     Flags
```

Example 3: Summary view of the IS-IS segment routing prefix segment

```
supervisor@rtbrick>spine1: op> show isis segment-routing prefix-segment
Instance: default, Level: 1
  System                    Prefix                    SID     Flags
  isr1                      192.168.1.1/32              100   Node
  isr1                      192:168:1::1/128            102   Node
  isr6                      192.168.1.6/32              600   Node
  isr6                      192:168:1::6/128            601   Node
Instance: default, Level: 2
  System                    Prefix                     SID     Flags
  isr1                      192.168.1.1/32               100   Node
  isr1                      192.168.1.6/32               600   Re-advertisement,
Node
  isr1                      192:168:1::1/128             102   Node
  isr1                      192:168:1::6/128             601   Re-advertisement,
Node
  isr6                      192.168.1.1/32               100   Re-advertisement,
Node
  isr6                      192.168.1.6/32               600   Node
  isr6                      192:168:1::1/128             102   Re-advertisement,
Node
  isr6                      192:168:1::6/128             601   Node
```

Example 4: Summary view of the IS-IS segment routing adjacency-segment

```
supervisor@S1-STD-17-1703: cfg> show isis segment-routing adjacency-segment
Instance: default, Level: 1, No adjacency-segments
Instance: default, Level: 2, No adjacency-segments
```

IS-IS SPF

Syntax:

show isis spf <option>

Option	**Description**
result	Displays a summary of the IS-IS SPF results
result <instance>	Displays a summary of the IS-IS SPF results for the specified instance

Option	Description
result [level-1/level2]	Displays a summary of the IS-IS SPF results for the specified level.

Example 1: Summary view of the IS-IS SPF result

```
supervisor@rtbrick>spine1: op> show isis spf result
Instance: default, Level: 1
  Destination Node          Metric     Neighbor Node         Interface       Nexthop   Address
  1921.6800.1001.00               0                          local
  1921.6800.1006.00          1000000   1921.6800.1006.00     ifl-0/0/2/0     IPv6      fe80::7847:eff:fec0:0
                                       1921.6800.1006.00     ifl-0/0/2/1     IPv6      fe80::7847:e00:1c0:0
                                       1921.6800.1006.00     ifl-0/0/2/0     IPv4      16.0.1.6
                                       1921.6800.1006.00     ifl-0/0/2/1     IPv4      16.0.2.6
  1921.6800.1006.01          1000000                           local
Instance: default, Level: 2
  Destination Node          Metric     Neighbor Node         Interface         Nexthop   Address
  1921.6800.1001.00               0                            local
  1921.6800.1006.00          1000000   1921.6800.1006.00     ifl-0/0/2/0      IPv6      fe80::7847:eff:fec0:0
                                       1921.6800.1006.00     ifl-0/0/2/1      IPv6      fe80::7847:e00:1c0:0
                                       1921.6800.1006.00     ifl-0/0/2/0      IPv4      16.0.1.6
                                       1921.6800.1006.00     ifl-0/0/2/1      IPv4      16.0.2.6
  1921.6800.1006.01          1000000                           local
```

Example 2: Summary view of the IS-IS SPF result for level-1

```
supervisor@rtbrick>spine1: op> show isis spf result level-1
Instance: default, Level: 1
  Destination Node          Metric       Neighbor Node         Interface         Nexthop   Address
  1921.6800.1001.00               0                            local
  1921.6800.1006.00          1000000     1921.6800.1006.00     ifl-0/0/2/0    IPv6      fe80::7847:eff:fec0:0
                                         1921.6800.1006.00     ifl-0/0/2/1    IPv6      fe80::7847:e00:1c0:0
                                         1921.6800.1006.00     ifl-0/0/2/0    IPv4      16.0.1.6
                                         1921.6800.1006.00     ifl-0/0/2/1    IPv4      16.0.2.6
  1921.6800.1006.01          1000000                           local
```

Example 3: Summary view of the IS-IS SPF result for level-2

```
supervisor@rtbrick>spine1: op> show isis spf result level-2
Instance: default, Level: 2
  Destination Node          Metric       Neighbor Node         Interface         Nexthop   Address
  1921.6800.1001.00               0                            local
  1921.6800.1006.00          1000000     1921.6800.1006.00     ifl-0/0/2/0       IPv6
fe80::7847:eff:fec0:0
                                         1921.6800.1006.00     ifl-0/0/2/1       IPv6
fe80::7847:e00:1c0:0
                                         1921.6800.1006.00     ifl-0/0/2/0       IPv4      16.0.1.6
                                         1921.6800.1006.00     ifl-0/0/2/1    IPv4      16.0.2.6
  1921.6800.1006.01          1000000                           local
```

Example 4: Summary view of the IS-IS SPF result of a specific instance for level-1

```
supervisor@rtbrick>spine1: op> show isis spf result instance default level-1
Instance: default, Level: 1
  Destination Node          Metric       Neighbor Node         Interface         Nexthop   Address
  1921.6800.1001.00               0                            local
  1921.6800.1006.00          1000000     1921.6800.1006.00     ifl-0/0/2/0       IPv6
fe80::7847:eff:fec0:0
```

```
                                          1921.6800.1006.00     ifl-0/0/2/1       IPv6
fe80::7847:e00:1c0:0
                                          1921.6800.1006.00     ifl-0/0/2/0       IPv4      16.0.1.6
                                          1921.6800.1006.00     ifl-0/0/2/1       IPv4      16.0.2.6
  1921.6800.1006.01          1000000                            local
```

Example 5: Summary view of the IS-IS SPF result of a specific instance for level-2

```
supervisor@rtbrick>spine1: op> show isis spf result instance default level-2
Instance: default, Level: 2
  Destination Node          Metric        Neighbor Node         Interface         Nexthop   Address
  1921.6800.1001.00                0                            local
  1921.6800.1006.00          1000000      1921.6800.1006.00     ifl-0/0/2/0       IPv6
fe80::7847:eff:fec0:0
                                          1921.6800.1006.00     ifl-0/0/2/1       IPv6
fe80::7847:e00:1c0:0
                                          1921.6800.1006.00     ifl-0/0/2/0       IPv4      16.0.1.6
                                          1921.6800.1006.00     ifl-0/0/2/1       IPv4      16.0.2.6
  1921.6800.1006.01          1000000                            local
```

IS-IS Clear Commands

Clear commands allow to reset operational states.

IS-IS Interface

Syntax:

clear isis interface <option> ...

Option	Description
statistics	Clears the statistics of all IS-IS interfaces.

Example: The example below shows how to clear IS-IS interface statistics.

```
supervisor@rtbrick>spine1: op> clear isis interface statistics
```

IS-IS Neighbor

Syntax:

clear isis neighbor <option> ...

Option	Description
neighbor	Clears neighbors of the default instance

Option	Description
neighbor instance <instance_name>	Clears neighbors of the specified instance
neighbor instance <instance_name> interface <interface-name>	Clears the specified interface of a specified neighbor instance

Example: The example below shows how to clear neighbors of the specified instance

```
supervisor@rtbrick>spine1: op> clear isis neighbor instance default
```

To access the Operational State API that corresponds to this CLI, click here.

IS-IS Database

Syntax:

clear isis database <option> ...

Option	Description
Instance	Clears neighbors of the default instance
level-1	Clears area level-1
level-2	Clears area level-2
lsp	Clears IS-IS database for the specific LSP ID
System	Clears the LSDB system ID

Example: The example below shows how to clear the database instance

```
supervisor@rtbrick>spine1: op> clear isis database instance default
```

The LSDB data is cleared for instance default

2.7. OSPFv2/v3

2.7.1. OSPF Overview

OSPF (Open Shortest Path First) is an Interior Gateway Protocol that distributes routing information within a single Autonomous System (AS) in an IP network. OSPF is a link-state routing protocol that uses link-state information to form a routing table and exchange the routing information with the neighbors.

RtBrick FullStack (RBFS) supports OSPF version 2 (OSPFv2) and OSPF version 3 (OSPFv3), including authentication, LDP-IGP sync, and redistribution policy. RBFS does not support OSPFv3 Virtual Link.

OSPF routers flood LSAs (link-state advertisements) to all other routers in an autonomous system. Routers generate routing tables using the information received from the LSAs and calculate the best path to other routers in the network. OSPF uses the Dijkstra (Shortest Path First) algorithm to calculate the best path.

LSAs contain local state information such as interfaces and the reachability of neighbors. Other routers, which receive this information as LSAs, build their LSDB (link-state database) using this information. In an OSPF network, all routers build and maintain information about the topology of that network.

OSPFv3

OSPF Version 3 (OSPFv3) is a modified version of OSPF and it provides support for the OSPF routing protocol within an IPv6 network. As such, it provides support for IPv6 addresses and prefixes. It retains most of the structure and functions in OSPFv2 (for IPv4) with a few changes.

OSPFv3 differs from OSPFv2 in the following ways:

- OSPFv3 runs on IPv6, which is based on links rather than network segments
- OSPFv3 does not depend on IP addresses
- OSPFv3 packets and the LSA format have the following changes:

    OSPFv3 router LSAs and network LSAs do not contain IP addresses, which are advertised by Type 8 LSAs and Type 9 LSAs

- In OSPFv3, information about the flooding scope is added in the LSA Type field

OSPFv3 stores or floods unidentified packets, whereas OSPFv2 discards

- OSPFv3 supports multi-process on a link with instance ID
- OSPFv3 uses IPv6 link-local addresses for forwarding

OSPFv3 Instance ID

One of the advancements of OSPFv3 over OSPFv2 is the use of the instance ID. This instance ID is an 8-bit field within the OSPFv3 header.

The original intent for the instance ID was to support multiple instances of OSPFv3 to run on the same interface. In this way, you can manipulate which routers on a particular segment are allowed to form adjacencies. You could use an instance number of 0 through 255 to distinguish between the different OSPFv3 instances.

However, within RFC 5838, the instance ID was re-purposed to be used to support address families (AFs) with OSPFv3. The default instance of 0 is used if no other instance is defined. However, specific ranges of the instance ID map to specific AFs. According to the RFC, these ranges are:

- Instance ID 0 to 31 — IPv6 unicast AF
- Instance ID 32 to 63 — IPv6 multicast AF
- Instance ID 64 to 95 — IPv4 unicast AF
- Instance ID 96 to 127 — IPv4 multicast AF
- Instance ID 128 to 255 — Unassigned

When using IPv4 unicast or IPv6 unicast, the allowed values for the command are between 0 and 31.

RBFS only supports IPv6 unicast addresses ranging from 0 to 31.

Understanding OSPF Areas

OSPF allows for a logical partition of the autonomous system by dividing it into areas. This logical partitioning helps to limit the flooding of link-state updates within an area.

An OSPF Autonomous System can be maintained as a single-area network or can be divided as a multi-area network. In a single area AS, the topology provides link-state information of routers in the entire autonomous system.

In a multi-area AS, the topology provides the link-state information of routers belonging to that particular area, not about routers in other areas in the autonomous system. Within an area, all OSPF routers maintain separate databases which are identical.

In a multi-area OSPF network, all areas are connected to the backbone area, known as Area 0.

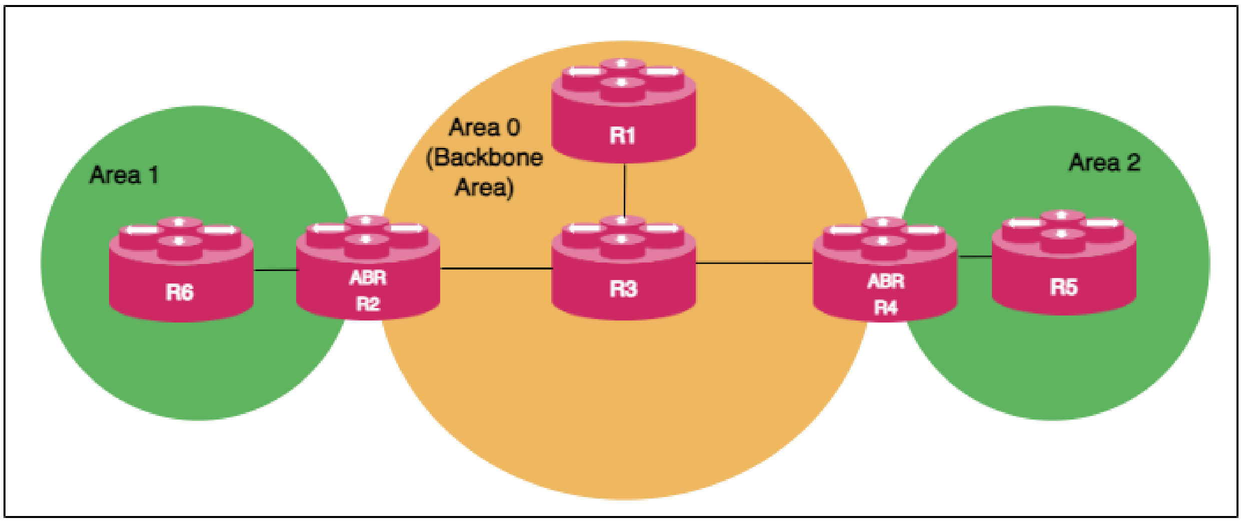
Backbone Area

The backbone area, also known as Area 0, is connected to all other areas in an OSPF network. The backbone area, which acts as a central point of communication, receives LSAs from other areas and disseminates the same to other areas.

Area Border Router

Routers that connect one or more areas with the backbone area are called Area Border Router (ABR). One interface of the ABR is connected to the backbone, while other interfaces are connected to other areas. ABRs, which belong to multiple areas in an OSPF network, maintain separate LSDBs for each area that they are connected to.

The following OSPF architectural diagram shows a simple OSPF network that is divided into areas. Area 1 and Area 2 are connected to the backbone area (Area 0) through the ABRs. Area 1 and Area 2 are not directly connected. They receive link state advertisements from each other from Area 0 which acts as the central point of communication for all other areas.



Autonomous System Boundary Router

ASBR (Autonomous System Boundary Router) serves as a gateway router to the OSPF autonomous system. ASBR can operate multiple protocols and work with other autonomous system routers that run other interior gateway protocols such as EIGRP, IS-IS, i-BGP, and so on. ASBR can import and translate different protocol routes into OSPF through the redistribution mechanism.

OSPF DR and BDR Election

An OSPF network chooses one router as a Designated Router (DR) and another as a Backup Designated Router (BDR) for a broadcast network.

DR acts as a central point of communication by receiving and distributing topology information. BDR takes over the role of DR if the DR fails. Routers in an OSPF network do not directly exchange routing information with each other. Instead, every router in the network updates routing information only with DR and BDR. DR, in turn, distributes the topology information with all other routers. This mechanism reduces network traffic significantly. OSPF chooses one router as DR and another router as BDR based on the following criteria:

- The router with the highest priority value becomes the designated router and the router with the second highest priority value becomes the BDR. You can define the priority values for routers during the interface configuration.
- If multiple routers have the same highest priority value, then the router with the highest router ID is elected as DR and the router with the second highest router ID value becomes the BDR.

You can choose a priority value from the range 0 - 255. Routers with the priority value '0' do not participate in the DR or BDR election.

Supported OSPF Standards

RBFS supports the following RFCs, which define standards for OSPFv2 and OSPFv3.

- RFC 2328, OSPF Version 2
- RFC 5340, OSPF for IPv6
- RFC 5709, OSPFv2 HMAC-SHA Cryptographic Authentication
- RFC 7166, Supporting Authentication Trailer for OSPFv3

- RFC 8665, OSPF Extensions for Segment Routing
- RFC 8666, OSPFv3 Extensions for Segment Routing

RFC and draft compliance are partial except as specified.

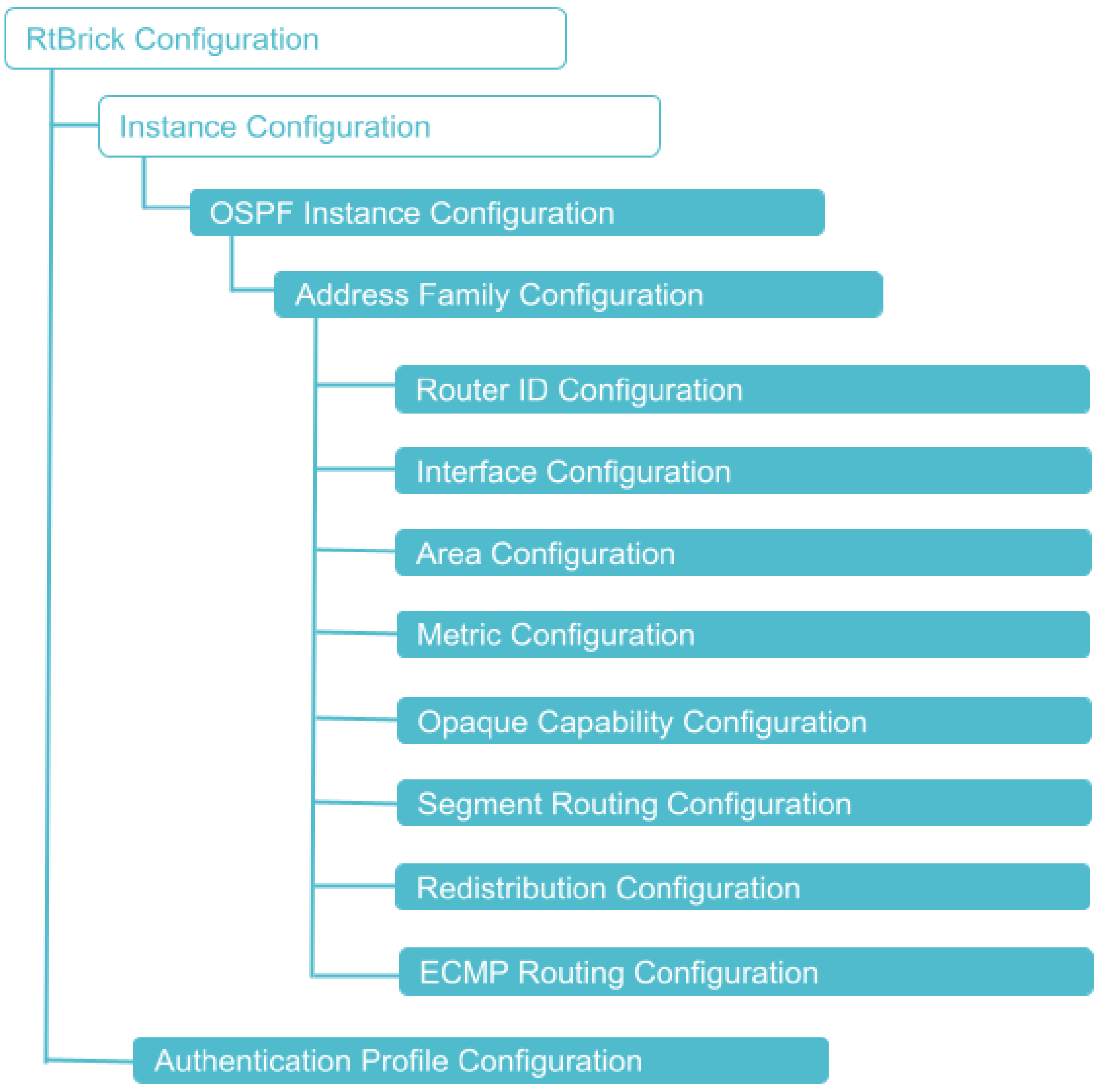
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

2.7.2. OSPF Configuration

Configuration Hierarchy

The diagram illustrates the OSPF configuration hierarchy. All OSPF configuration is performed within an instance, for example, the default instance or a VPN service instance. The OSPF instance configuration hierarchy includes parameters that are generic to the respective OSPF instance. The sub-hierarchies include parameters that are specific to redistribution or authentication.


RtBrick Configuration
Instance Configuration
OSPF Instance Configuration
Address Family Configuration
Router ID Configuration
Interface Configuration
Area Configuration
Metric Configuration
Opaque Capability Configuration
Segment Routing Configuration
Redistribution Configuration
ECMP Routing Configuration
Authentication Profile Configuration

Configuration Syntax and Commands

The following sections describe the OSPF configuration syntax and commands.

OSPF Instance Configuration

At this configuration hierarchy, you can configure an OSPF instance.

Syntax:

set instance <instance-name> **protocol ospf**

Attribute	Description
<instance-name>	Name of the OSPF instance.

OSPF Address Family Configuration

At this configuration level, you configure the OSPF protocol address family.

You must configure the OSPF address family on an OSPF instance before configuring other supported OSPF features.

Syntax

set instance <instance-name> **protocol ospf address-family** <afi>

Attribute	Description
<instance-name>	Name of the instance
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.

The following example shows the OSPF address family (IPv4) configuration.

Example 1: OSPF Instance Address Family IPv6 Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4
{
  "rtbrick-config:ipv4": {
    "router-id": "198.51.100.10",
    "area": [
      {
        "area-id": "0.0.0.0",
        "interface": [
          {
            "name": "ifl-0/0/0/1",
            "network-type": "p2p"
          },
          {
            "name": "lo-0/0/0/1"
          }
        ]
      }
    ]
  }
}
<...>
```

The following example shows the OSPF address family (IPv6) configuration.

Example 2: OSPF Instance Address Family IPv6 Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6
{
  "rtbrick-config:ipv6": {
    "instance-id": [
      {
        "instance-id": 0,
        "router-id": "192.168.0.10",
        "metric": 1000,
        "segment-routing": {
          "status": "enable",
          "srgb": {
            "base": 1000,
            "range": 1000
          }
        },
        "redistribute": [
          {
            "source": "bgp",
            "metric": 2000
          },
          {
            "source": "direct",
            "policy": "ospf_policy_1"
          }
        ],
        "area": [
          {
            "area-id": "0.0.0.0",
            "metric": 1000,
            "interface": [
              {
                "name": "ifl-0/0/0/1",
                "authentication-profile": "AUTH_PROFILE_SHA_512_1"
              },
              {
                "name": "ifl-0/0/0/100",
                "metric": 20000,
                "network-type": "p2p",
                "authentication-profile": "AUTH_PROFILE_SHA_512_1"
              },
              {
                "name": "ifl-0/0/1/1",
                "metric": 40000,
                "network-type": "p2p",
                "authentication-profile": "AUTH_PROFILE_SHA_512_1"
              },
              {
                "name": "ifl-0/0/1/100",
                "metric": 30000,
                "authentication-profile": "AUTH_PROFILE_SHA_512_1"
<...>
```

OSPF Router ID Configuration

The router ID is an IP address that OSPF uses to identify a device on the network. The router ID should be configured under the address family hierarchy.

Syntax

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
router-id <ipv4-address>	The router ID of the routing instance. It is recommended to specify the router ID.

Example 1: OSPF IPv4 Router Identifier Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf
address-family ipv4 router-id
{
  "rtbrick-config:router-id": "192.168.0.10"
}
supervisor@rtbrick>SPINE01: cfg>
```

Example 2: OSPF IPv6 Router Identifier Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf
address-family ipv6 instance-id 0 router-id
{
  "rtbrick-config:router-id": "192.168.0.10"
}
supervisor@rtbrick>SPINE01: cfg>
```

OSPF Hostname Configuration

Identifying routers through Open Shortest Path First (OSPF) hostnames can simplify network management as they are easier to remember than router IDs.

Syntax

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.

Attribute	Description
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
hostname <hostname>	The hostname of the routing instance.

Example 1: OSPF IPv4 Router Identifier Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf
address-family ipv4 hostname
{
  "rtbrick-config:hostname": "multiservice-edge"
}
supervisor@rtbrick>SPINE01: cfg>
```

Example 2: OSPF IPv6 Router Identifier Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf
address-family ipv6 instance-id 0 hostname
{
  "rtbrick-config:hostname": "multiservice-edge"
}
supervisor@rtbrick>SPINE01: cfg>
```

OSPF Area Configuration

A particular area is defined by its area ID.

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
area <area-id>	Specifies the OSPF area ID.
area <area-id> metric	Area scope metric. Range: 1 - 65535. Default: 10000.

Attribute	Description
area <area-id> area-type stub	A stub area is an area through which or into which AS external advertisements are not flooded instead type-3 LSA advertise with a default route.
area <area-id> area-type totally-stub	Totally stub area is an area in which type-3 LSAs are not allowed instead type-3 LSAs advertise with a default route.
area <area-id> authentication-profile <authentication-profile>	Specifies the authentication profile name used to create an attachment point at the area level.
area <area-id> no-authentication-check <enable>	When enabled, OSPF packets received here will not undergo authentication validation, even if the user has enabled authentication. However, OSPF will continue to send authenticated packets from this interface.

Example 1: OSPF IPv4 Area Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 area 0.0.0.0
{
  "rtbrick-config:area": [
    {
      "area-id": "0.0.0.0",
      "interface": [
        {
          "name": "ifl-0/0/0/1",
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/0/100",
          "metric": 20000,
          "network-type": "p2p",
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/1/1",
          "metric": 40000,
          "network-type": "p2p",
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/1/100",
          "metric": 30000,
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/4/1",
          "metric": 60000
        },
        {
          "name": "lo-0/0/0/1"
        },
        {
          "name": "lo-0/0/0/2"
        }
      ]
    }
  ]
}
```

Example 2: OSPF IPv6 Area Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6 instance-id 0
area 0.0.0.0
{
  "rtbrick-config:area": [
    {
      "area-id": "0.0.0.0",
      "interface": [
        {
          "name": "ifl-0/0/0/1",
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/0/100",
          "metric": 20000,
          "network-type": "p2p",
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/1/1",
          "metric": 40000,
          "network-type": "p2p",
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/1/100",
          "metric": 30000,
          "authentication-profile": "AUTH_PROFILE_SHA_512_1"
        },
        {
          "name": "ifl-0/0/4/1",
          "metric": 60000
        },
        {
          "name": "lo-0/0/0/1"
        },
        {
          "name": "lo-0/0/0/2"
        }
      ]
    }
  ]
}
```

OSPF Interface Configuration

Enable OSPF protocol on the router interfaces.

Syntax:

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
area <area-id> interface <interface-name> ldp-synchronization enable	Enables LDP OSPF Synchronization. By default, LDP OSPF synchronization is disabled.
metric <metric>	Specify the metric value of an OSPF interface.
network-type <broadcast | p2p>	broadcast - Sets the network type to broadcast; p2p - Sets the network type to point-to-point. By default, the network-type is broadcast.
router-priority <router-priority>	Sets the router priority for an interface. Allowed range: 0 - 255, Default: 1. Routers with priority value '0' do not participate in the DR or BDR election.
segment-routing index	Sets the prefix segment identifier (SID) index for the specified interface.
timer <hello | dead | wait>	Interface timer for configuring hello, dead, and wait timers. • **hello**: Sets interval time for sending hello packets to a neighbor and this time is identical on OSPF neighbor routers. Default: 10 seconds. • **dead**: Sets interval time within which if the interface does not receive any hello packet from its neighbor, the interface comes to know that the neighbor is down. Default: 40 seconds. • **wait**: Sets wait time to delay to trigger the DR/BDR election. It cannot be more than the time set for the dead interval. Default: 40 seconds.

Attribute	Description
mtu-ignore enable	If there is an MTU mismatch on both sides of the link where OSPF runs, the OSPF adjacency will not come up as the MTU value carried in the Database Description (DBD) packets. To avoid MTU validation in the Database Description (DBD) packets, configure mtu-ignore command. By default, it is disabled.
authentication <authentication-profile>	Specifies the authentication profile name used to create an attachment point at the interface level.
no-authentication-check <enable>	When enabled, OSPF packets received here will not undergo authentication validation at the interface level, even if the user has enabled authentication.

If an authentication profile is attached to an interface and an area, the authentication profile attached to the interface takes priority.

Example 1: OSPF IPv4 Interface Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 area 0.0.0.0
interface
{
  "rtbrick-config:interface": [
    {
      "name": "ifl-0/0/0/1",
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/0/100",
      "metric": 20000,
      "network-type": "p2p",
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/1/1",
      "metric": 40000,
      "network-type": "p2p",
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/1/100",
      "metric": 30000,
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/4/1",
      "metric": 60000
    },
    {
      "name": "lo-0/0/0/1"
    },
    {
      "name": "lo-0/0/0/2"
    }
  ]
}
```

Example 2: OSPF IPv6 Interface Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6 instance-id 0
area 0.0.0.0 interface
{
  "rtbrick-config:interface": [
    {
      "name": "ifl-0/0/0/1",
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/0/100",
      "metric": 20000,
      "network-type": "p2p",
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/1/1",
      "metric": 40000,
      "network-type": "p2p",
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/1/100",
      "metric": 30000,
      "authentication-profile": "AUTH_PROFILE_SHA_512_1"
    },
    {
      "name": "ifl-0/0/4/1",
      "metric": 60000
    },
    {
      "name": "lo-0/0/0/1"
    },
    {
      "name": "lo-0/0/0/2"
    }
  ]
}
```

OSPF Metric Configuration

Metric is the cost that OSPF uses to calculate and identify the best paths to other routers.

Syntax

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.

Attribute	Description
metric <metric>	OSPF address-family metric. Allowed range: 1 - 65535. Default: 10000. If you configure the metric at the address family, it will be applicable to the configured areas of the address-family. If you configure a metric for an area, this configured metric value will take precedence over the address-family metric configurations of this area. If you specify a metric value for an area on an interface will override any area and address-family metric configurations for this area.

Example 1: OSPF IPv4 Metric Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 area 0.0.0.0
metric
{
  "rtbrick-config:metric": 1000
}
```

Example 2: OSPF IPv6 Metric Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6 instance-id 0
area 0.0.0.0 metric
{
  "rtbrick-config:metric": 1000
}
```

OSPF Opaque Capability Configuration

Enables opaque link-state advertisements. Routers in the OSPF network can receive and advertise Type-9, Type-10 and Type-11 opaque LSAs.

Syntax

set instance <instance-name> **protocol ospf address-family ipv4** <attribute> <value>

Attribute	Description
opaque-capability <enable | disable>	Enable or disable opaque LSA advertisement and reception. Set as 'enable' to enable the router to receive and advertise opaque LSAs.

Example: OSPF Opaque Capability Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 opaque-capability
{
  "rtbrick-config:opaque-capability": "enable"
}
```

Segment Routing Configuration

Enable segment routing for OSPF. For configuring segment routing, you must enable the opaque capability by defining it as 'true'. For information, see the section: "Opaque Capability Configuration".

Syntax

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
segment-routing srgb base <value>	Specifies the segment routing global block (SRGB) in source packet routing. SRGB is used for prefix SIDs. Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. It is recommended to assign label values outside of these reserved ranges to avoid conflicts.
segment-routing srgb range <value>	OSPF system range of labels from the base label.

Attribute	Description
segment-routing status <disable | enable>	Enable or disable the segment routing feature. By default, the status is disabled.

Example 1: OSPF IPv4 Segment routing Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 segment-
routing
{
  "rtbrick-config:segment-routing": {
    "status": "enable",
    "srgb": {
      "base": 1000,
      "range": 1000
    }
  }
}
```

Example 2: OSPF IPv6 Segment routing Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6 instance-id 0
segment-routing
{
  "rtbrick-config:segment-routing": {
    "status": "enable",
    "srgb": {
      "base": 1000,
      "range": 1000
    }
  }
}
```

OSPF Redistribution Configuration

Enable route redistribution for the routes originating from other sources or protocols such as BGP, Direct, IPoE, IS-IS, PPP, and Static.

Syntax

set instance <instance-name> **protocol ospf** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.

Attribute	Description
redistribute <protocol>	Specifies the source protocol from which the routes are to be redistributed. The available options include BGP, Direct, IPoE, IS-IS, PPP, and Static.
metric <metric>	Specifies the metric value for the redistributed routes
metric-type <type 1 | type 2>	Specifies the external metric type for the redistributed routes.
policy	Specifies the name of the policy map. The redistribute attach point allows routes from other sources to be advertised by OSPFv2.

Example 1: OSPF IPV4 BGP Redistribution Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 redistribute
bgp
{
  "rtbrick-config:redistribute": [
    {
      "source": "bgp",
      "metric": 2000
    }
  ]
}
```

Example 2: OSPF IPv6 Redistribution Policy

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6 instance-id 0
redistribute direct
{
  "rtbrick-config:redistribute": [
    {
      "source": "direct",
      "policy": "ospf_policy_1"
    }
  ]
}
```

ECMP Routing Configuration

ECMP (equal-cost multiple paths) routing is a mechanism in which routers forward packets to a destination using the multiple available best paths. This mechanism can increase network bandwidth substantially by load-balancing traffic through multiple best paths.

Syntax

set instance <instance-name> **protocol ospf address-family** <afi> <attribute> <value>

Attribute	Description
<afi>	Specifies the Address family identifier (AFI), either IPv4 or IPv6.
<afi> instance-id <instance-id>	Specifies the instance ID. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
<max-load-balance>	Maximum number of equal-cost multiple paths to be calculated for load balancing. Default: 16. Allowed range: 1 - 255.

Example: OSPF IPv4 ECMP Routing Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv4 max-load-
balance
{
  "rtbrick-config:max-load-balance": 100
}
```

Example: OSPF IPv6 ECMP Routing Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ospf address-family ipv6 instance-id 0
max-load-balance
{
  "rtbrick-config:max-load-balance": 100
}
```

OSPF Authentication Configuration

OSPF supports the secure exchange of routing updates through authentication. You can enable authentication by attaching an authentication profile at the area or interface level. OSPF allows multiple keys to be attached to prevent session interruption.

The table below shows the authentication types supported by OSPFv2 and v3.

Authentication Type	OSPFv2	OSPFv3
Clear Text	Yes	No

Authentication Type	OSPFv2	OSPFv3
HMAC-SHA-1	Yes	Yes
HMAC-SHA-256	Yes	Yes
HMAC-SHA-384	Yes	Yes
HMAC-SHA-512	Yes	Yes
MD5	Yes	No

To authenticate OSPF, there must be a global authentication profile present.

Configuring an Authentication Profile

set authentication-profile <attribute> <value>

Attribute	Description
<name>	Specifies the authentication profile name.
<name> key <key-id>	Specifies the message digest key identifier to be used by the neighboring routers for the OSPF password authentication. Allowed range: 1 - 255.
<name> key <key-id> type <auth-type>	Specifies the type of authentication that is being used, such as MD5, HMAC-SHA-1, and others.
<name> key <key-id> plain-text <text>	Specifies the password in plain text format.
<name> key <key-id> encrypted-text <text>	Specifies the password in an encrypted text format.
<name> key prefer-key-id <key-id>	Preferred key-id configuration will be used while sending out the packet with the specified key.

- When an authentication profile is available, you can configure an authentication attachment point at the area or interface level.
- When an authentication profile contains multiple key-IDs, and the preferred key-ID is not configured, the packet is sent using the highest key-ID.

In the example below, the authentication profile "auth-profile1" has md5, hmac-sha-1, and clear-text enabled. The preferred key-id being 20, the hmac-sha-1 method will be used for authentication.

```
    "rtbrick-config:instance": [
      {
        "name": "default",
        "address-family": [
          {
            "afi": "ipv4",
            "safi": "unicast"
          },
          {
            "afi": "ipv6",
            "safi": "unicast"
          }
        ],
        "protocol": {
          "ospf": {
            "address-family": {
              "ipv6": {
                "instance-id": [
                  {
                    "instance-id": 0,
                    "router-id": "192.168.0.10",
                    "max-load-balance": 100,
                    "metric": 1000,
                    "segment-routing": {
                      "status": "enable",
                      "srgb": {
                        "base": 1000,
                        "range": 1000
                      }
                    },
                    "redistribute": [
                      {
                        "source": "bgp",
                        "metric": 2000
                      },
                      {
                        "source": "direct",
                        "policy": "ospf_policy_1"
                      }
                    ],
                    "area": [
                      {
                        "area-id": "0.0.0.0",
                        "metric": 1000,
                        "interface": [
                          {
                            "name": "ifl-0/0/0/1",
                            "authentication-profile": "AUTH_PROFILE_SHA_512_1"
                          },
                          {
                            "name": "ifl-0/0/0/100",
                            "metric": 20000,
                            "network-type": "p2p",
                            "authentication-profile": "AUTH_PROFILE_SHA_512_1"
```

2.7.3. OSPF Operational Commands

OSPF Show Commands

OSPF Summary

Displays the OSPF protocol summary information.

Syntax:

show ospf summary <options>

Option	Description
-	Without any option, the command displays the information for all instances and address families.
instance <instance-name>	OSPF summary information for the given instance.
instance <instance-name> <afi>	OSPF summary information for the specified instance and address family. Supported AFI values are 'ipv4' and 'ipv6'.
<afi> instance <instance-name>	OSPF summary information for the specified address family and instance. Supported AFI values are 'ipv4' and 'ipv6'.

Example: OSPF summary for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf summary
Global Information:
  Neighbor State Information:
    Full       : 12
    Loading    : 0
    Exchange   : 0
    ExStart    : 0
    TwoWay     : 2
    Init       : 0
    Attempt    : 0
    Down       : 0
Instance: default, Address family: ipv4
  General information:
    Router ID: 192.168.0.10, Area count: 1, Flood interval: 1000ms
    Opaque capability: True, Segment routing capability: True
    Flags: -|-|-|-|-, Cost: 10000
    SPF initial delay: 50ms, SPF short delay: 200ms, SPF long delay: 5000ms
  Area: 0.0.0.0
    Interface count: 7
      Interface: ifl-0/0/0/1
        Address: 12.0.0.1, State: BDR, Type: broadcast, Priority: 1
        Designated router: 12.0.0.2, Backup designated router: 12.0.0.1
        Hello interval: 10s, Dead interval: 40s
        Cost: 10000, MTU: 1500
      Interface: ifl-0/0/0/100
        Address: 12.1.0.1, State: P2P, Type: p2p, Priority: 1
        Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
        Hello interval: 10s, Dead interval: 40s
        Cost: 20000, MTU: 1500
      Interface: ifl-0/0/1/1
        Address: 12.2.0.0, State: P2P, Type: p2p, Priority: 1
        Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
        Hello interval: 10s, Dead interval: 40s
        Cost: 40000, MTU: 1500
      Interface: ifl-0/0/1/100
        Address: 12.3.0.1, State: BDR, Type: broadcast, Priority: 1
        Designated router: 12.3.0.2, Backup designated router: 12.3.0.1
        Hello interval: 10s, Dead interval: 40s
        Cost: 30000, MTU: 1500
      Interface: ifl-0/0/4/1
        Address: 11.0.0.1, State: DROther, Type: broadcast, Priority: 1
        Designated router: 11.0.0.5, Backup designated router: 11.0.0.4
        Hello interval: 10s, Dead interval: 40s
        Cost: 60000, MTU: 1500
      Interface: lo-0/0/0/1
        Address: 192.168.0.10, State: P2P, Type: p2p, Priority: 1
        Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
        Hello interval: 10s, Dead interval: 40s
        Cost: 10000, MTU:
      Interface: lo-0/0/0/2
        Address: 192.168.0.11, State: P2P, Type: p2p, Priority: 1
        Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
        Hello interval: 10s, Dead interval: 40s
        Cost: 10000, MTU:
Instance: default, Address family: ipv6, Instance ID: 0
  General information:
    Router ID: 192.168.0.10, Area count: 1, Flood interval: 1000ms
    Opaque capability: True, Segment routing capability: True
    Flags: -|-|-|-|-, Cost: 10000
    SPF initial delay: 50ms, SPF short delay: 200ms, SPF long delay: 5000ms
  Area: 0.0.0.0
    Interface count: 7
      Interface: ifl-0/0/0/1
        Address: fe80::7810:99ff:fec0:0, State: BDR, Type: broadcast, Priority: 1
        Designated router: 192.168.0.20, Backup designated router: 192.168.0.10
        Hello interval: 10s, Dead interval: 40s
        Cost: 10000, MTU: 1500
      Interface: ifl-0/0/0/100
        Address: fe80::65:7810:99ff:fec0:0, State: P2P, Type: p2p, Priority: 1
        Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
        Hello interval: 10s, Dead interval: 40s
        Cost: 20000, MTU: 1500
      Interface: ifl-0/0/1/1
        Address: fe80::7810:99ff:fec0:1, State: P2P, Type: p2p, Priority: 1
        Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
        Hello interval: 10s, Dead interval: 40s
        Cost: 40000, MTU: 1500
      Interface: ifl-0/0/1/100
        Address: fe80::65:7810:99ff:fec0:1, State: BDR, Type: broadcast, Priority: 1
        Designated router: 192.168.0.20, Backup designated router: 192.168.0.10
        Hello interval: 10s, Dead interval: 40s
        Cost: 30000, MTU: 1500
      Interface: ifl-0/0/4/1
        Address: fe80::7810:99ff:fec0:4, State: DROther, Type: broadcast, Priority: 1
```

```
          Designated router: 192.168.0.50, Backup designated router: 192.168.0.40
          Hello interval: 10s, Dead interval: 40s
          Cost: 60000, MTU: 1500
        Interface: lo-0/0/0/1
          Address: 192:168::10, State: P2P, Type: p2p, Priority: 1
          Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
          Hello interval: 10s, Dead interval: 40s
          Cost: 10000, MTU: 0
        Interface: lo-0/0/0/2
          Address: 192:168::11, State: P2P, Type: p2p, Priority: 1
          Designated router: 0.0.0.0, Backup designated router: 0.0.0.0
          Hello interval: 10s, Dead interval: 40s
          Cost: 10000, MTU: 0
<...>
```

To access the Operational State API that corresponds to this CLI, click here.

OSPF Hostname

Displays the OSPF hostname information.

Syntax:

show ospf hostname

Example: OSPF hostname for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf hostname
Instance                       AFI   Instance ID     Router ID               Hostname
default                        ipv4                  192.168.0.10            Router1
default                        ipv6  0               192.168.0.10            Router1
```

OSPF Interface

Displays OSPF interface information.

Syntax:

show ospf interface <options>

Option	Description
-	Without any option, the command displays the interface information for all instances and address families.

Option	Description
detail	Displays the detailed interface information.
<interface-name> detail	Displays detailed information for the specified interface. Also, for the specified interface, you can display interface information with filter options: detail, ipv4, and ipv4 detail.
instance <instance-name>	OSPF interface information for the given instance. Also, for the specified instance, you can display interface information with filter options: interface name, detail, and ipv4 detail.
instance <instance-name> <interface-name>	Displays information for a specified interface for a given instance.
instance <instance-name> <interface-name> detail	Displays detailed information for a specified interface for a given instance.
<afi>	Displays OSPF interface information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. You can display interface information for the specified address family using filters such as interface name, detail, and instance.

Example 1: OSPF interface information for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf interface
Instance: default, Address family: ipv4
  Interface            Area            IP Address                     State          Type       Cost
Priority DR             BDR             MTU
  ifl-0/0/0/1       0.0.0.0         12.0.0.1                       BDR            broadcast  10000    1
12.0.0.2        12.0.0.1        1500
  ifl-0/0/0/100     0.0.0.0         12.1.0.1                       P2P            p2p        20000    1
0.0.0.0         0.0.0.0         1500
  ifl-0/0/1/1       0.0.0.0         12.2.0.0                       P2P            p2p        40000    1
0.0.0.0         0.0.0.0         1500
  ifl-0/0/1/100     0.0.0.0         12.3.0.1                       BDR            broadcast  30000    1
12.3.0.2        12.3.0.1        1500
  ifl-0/0/4/1       0.0.0.0         11.0.0.1                       DROther        broadcast  60000    1
11.0.0.5        11.0.0.4        1500
  lo-0/0/0/1           0.0.0.0         192.168.0.10                   P2P            p2p        10000    1
0.0.0.0         0.0.0.0
  lo-0/0/0/2           0.0.0.0         192.168.0.11                   P2P            p2p        10000    1
0.0.0.0         0.0.0.0
Instance: default, Address family: ipv6, Instance ID: 0
  Interface            Area            IP Address                     State          Type       Cost
Priority DR             BDR             MTU
  ifl-0/0/0/1       0.0.0.0         fe80::7810:99ff:fec0:0         BDR            broadcast  10000    1
192.168.0.20    192.168.0.10    1500
  ifl-0/0/0/100     0.0.0.0         fe80::65:7810:99ff:fec0:0      P2P            p2p        20000    1
0.0.0.0         0.0.0.0         1500
  ifl-0/0/1/1       0.0.0.0         fe80::7810:99ff:fec0:1         P2P            p2p        40000    1
0.0.0.0         0.0.0.0         1500
  ifl-0/0/1/100     0.0.0.0         fe80::65:7810:99ff:fec0:1      BDR            broadcast  30000    1
192.168.0.20    192.168.0.10    1500
```

```
  ifl-0/0/4/1       0.0.0.0         fe80::7810:99ff:fec0:4         DROther        broadcast  60000    1
192.168.0.50    192.168.0.40    1500
  lo-0/0/0/1           0.0.0.0         192:168::10                    P2P            p2p        10000    1
0.0.0.0         0.0.0.0         0
  lo-0/0/0/2           0.0.0.0         192:168::11                    P2P            p2p        10000    1
0.0.0.0         0.0.0.0         0
```

Example 2: OSPF interface detailed information

```
supervisor@rtbrick>SPINE01: cfg> show ospf interface ifl-0/0/1/100 detail
Instance: default, Address family: ipv4
  Interface            Area            IP Address                     State          Type       Cost
Priority DR              BDR             MTU
  ifl-0/0/1/100     0.0.0.0         12.3.0.1                       BDR            broadcast  30000    1
12.3.0.2        12.3.0.1        1500
Instance: default, Address family: ipv6, Instance ID: 0
  Interface            Area            IP Address                     State          Type       Cost
Priority DR              BDR             MTU
  ifl-0/0/1/100     0.0.0.0         fe80::65:7810:99ff:fec0:1      BDR            broadcast  30000    1
192.168.0.20    192.168.0.10    1500
```

To access the Operational State API that corresponds to this CLI, click here.

OSPF Neighbor

Displays OSPF neighbor information.

Syntax:

show ospf neighbor <options>

Option	Description
-	Without any option, the command displays the neighbor information for all instances.
instance <instance-name>	OSPF neighbor information for the given instance.
area <area-id>	OSPF neighbor information for the given area.
detail	Displays the detailed neighbor information.
interface <interface-name>	Displays the neighbor information for a specified interface.
instance <instance-name> detail	Displays detailed OSPF neighbor information for the given instance.

Option	Description
instance <instance-name> interface <interface-name>	Displays OSPF neighbor information for the specified interface for the given instance.
instance <instance-name> interface <interface-name> detail	Displays detailed OSPF neighbor information for the specified interface for the given instance.
interface <interface-name> detail	Displays detailed neighbor information for a specified interface.
<afi>	Displays OSPF neighbor information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. You can display neighbor information for the specified address family using filters such as interface name, detail, and instance.
log	Logs neighbor event information.

Example: OSPF neighbor information for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf neighbor
Instance: default, Address family: ipv4
  Address                        Interface            Router          Area            State      Priority  DR
   BDR             Uptime          Expires
  12.0.0.2                       ifl-0/0/0/1       192.168.0.20    0.0.0.0         Full       1
12.0.0.2        12.0.0.1        0d:01h:38m:12s  33s
  12.2.0.1                       ifl-0/0/1/1       192.168.0.20    0.0.0.0         Full       1
0.0.0.0         0.0.0.0         0d:01h:38m:57s  33s
  11.0.0.3                       ifl-0/0/4/1       192.168.0.30    0.0.0.0         TwoWay     1
11.0.0.5        11.0.0.4        never           34s
  11.0.0.4                       ifl-0/0/4/1       192.168.0.40    0.0.0.0         Full       1
11.0.0.5        11.0.0.4        0d:01h:38m:11s  34s
  11.0.0.5                       ifl-0/0/4/1       192.168.0.50    0.0.0.0         Full       1
11.0.0.5        11.0.0.4        0d:01h:38m:16s  34s
  12.1.0.2                       ifl-0/0/0/100     192.168.0.20    0.0.0.0         Full       1
0.0.0.0         0.0.0.0         0d:01h:38m:57s  33s
  12.3.0.2                       ifl-0/0/1/100     192.168.0.20    0.0.0.0         Full       1
12.3.0.2        12.3.0.1        0d:01h:38m:12s  33s
Instance: default, Address family: ipv6, Instance ID: 0
  Address                        Interface            Router          Area            State      Priority  DR
   BDR             Uptime          Expires
  fe80::7845:9aff:fec0:0         ifl-0/0/0/1       192.168.0.20    0.0.0.0         Full       1
192.168.0.20    192.168.0.10    0d:01h:38m:12s  33s
  fe80::7845:9aff:fec0:1         ifl-0/0/1/1       192.168.0.20    0.0.0.0         Full       1
0.0.0.0         0.0.0.0         0d:01h:38m:52s  33s
  fe80::780d:96ff:fec0:4         ifl-0/0/4/1       192.168.0.30    0.0.0.0         TwoWay     1
192.168.0.50    192.168.0.40    never           34s
  fe80::7801:34ff:fec0:5         ifl-0/0/4/1       192.168.0.40    0.0.0.0         Full       1
192.168.0.50    192.168.0.40    0d:01h:38m:16s  34s
  fe80::7865:1aff:fec0:6         ifl-0/0/4/1       192.168.0.50    0.0.0.0         Full       1
192.168.0.50    192.168.0.40    0d:01h:38m:16s  34s
  fe80::65:7845:9aff:fec0:0      ifl-0/0/0/100     192.168.0.20    0.0.0.0         Full       1
0.0.0.0         0.0.0.0         0d:01h:38m:52s  33s
  fe80::65:7845:9aff:fec0:1      ifl-0/0/1/100     192.168.0.20    0.0.0.0         Full       1
192.168.0.20    192.168.0.10    0d:01h:38m:12s  33s
```

To access the Operational State API that corresponds to this CLI, click here.

OSPF Segment Routing

Displays OSPF segment routing information.

Syntax:

show ospf segment-routing <options>

Option	Description
-	Without any option, the command displays the neighbor information for all instances.
global-block	Displays Segment routing global block (SRGB) information
global-block instance <instance-name>	Displays Segment routing global block (SRGB) information for the specified instance
global-block instance <instance-name> <afi>	Displays Segment routing global block (SRGB) information for the specified instance address family. Supported AFI values are 'ipv4' and 'ipv6'.
prefix-segment	Displays the OSPF prefix segment information
prefix-segment <instance> Displays the OSPF prefix segments for the specified instance and address family	prefix-segment <instance-name> <afi>

Example: OSPF neighbor information for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf segment-routing

<...>
```

OSPF Database

Displays information from OSPF link-state database that contains data about link-state advertisements (LSAs).

Syntax:

show ospf database <options>

Option	Description
advertising-router <router-id>	Displays LSDB information for the specified advertising router.
advertising-router <router-id> detail	Displays the detailed LSDB information for the specified advertising router.
advertising-router <router-id> ls-id <ls-id>	Displays the LSDB information for the specified link-state ID for the advertising router.
advertising-router <router-id> ls-type external	Displays the LSDB information for the specified LSA type for the advertising router. Link-state advertisement types include external, network, router, and summary.
detail	Displays detailed information from LSDB.
instance <instance-name>	Displays OSPF database information for the given instance.
ls-id <ls-id>	OSPF database information for a specific link-state identifier.
ls-type <type>	OSPF database information for the specified link-state type. Link-state advertisement types include external, network, router and summary.
area <area-id>	Displays database information for the specified OSPF area.
area <area-id> advertising-router <router-id>	Displays LSDB information for the specified advertising router for a specified OSPF area.
area <area-id> detail	Displays detailed LSDB information for the specified OSPF area.
area <area-id> ls-id <ls-id>	Displays LSDB information for the specified link-state identifier for the specified OSPF area.

Option	Description
area <area-id> ls-type <type>	Displays LSDB information for the specified link-state type for the specified OSPF area. Link-state advertisement types include external, network, router, and summary.
instance <instance-name> advertising router <router-id>	Displays LSDB information for the specified advertising router for the given instance.
instance <instance-name> area <area-id>	Displays LSDB information for the specified area for the given instance.
instance <instance-name> ls-id <ls-id>	Displays LSDB information for the specified link-state identifier for the given instance.
instance <instance-name> ls-type <type>	Displays LSDB information for the specified type of the given instance. Link-state advertisement types include external, network, router and summary.
<afi>	Displays LSDB neighbor information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. You can display LSDB information for the specified address family using filters such as interface name, detail, and instance.

Example 1: OSPF database information for the default instance

```
supervisor@rtbrick>SPINE01: cfg> show ospf database
Instance: default, Address family: ipv4, Area: 0.0.0.0
  Type                    Link State ID   Advertising Router          Age        Sequence   Checksum
Cost
  Router                  192.168.0.10    192.168.0.10                578      0x80000008     0x262a
-
  Router                  192.168.0.20    192.168.0.20                579      0x80000009     0xddd7
-
  Router                  192.168.0.30    192.168.0.30                584      0x80000007     0x92be
-
  Network                 11.0.0.5        192.168.0.50                583      0x80000004     0xc9b1
-
  Network                 12.0.0.2        192.168.0.20                579      0x80000004     0x7962
-
  Network                 12.3.0.2        192.168.0.20                579      0x80000004     0xdc7a
-
  Network                 23.0.0.3        192.168.0.30                585      0x80000004     0x9917
-
  Summary-Network         24.0.1.0        192.168.0.20                623      0x80000004     0xbc36
15000
  Summary-Network         24.0.1.0        192.168.0.40                623      0x80000004     0x449a
15000
  Summary-Network         24.0.1.0        192.168.0.50                578      0x80000005      0x795
25000
Instance: default, Address family: ipv4
  Type                    Link State ID   Advertising Router          Age        Sequence   Checksum
Cost
```

```
  External                  200.0.1.0       192.168.0.60              619      0x80000004     0xba29
16777214
  External                  200.0.1.60      192.168.0.60              619      0x80000004     0x6047
16777214
  External                  200.0.1.61      192.168.0.60              619      0x80000004     0x5650
16777214
Instance: default, Address family: ipv6, Instance ID: 0, Area: 0.0.0.0
  Type                      Link State ID   Advertising Router        Age        Sequence   Checksum
Cost
  OSPFv3-Router             0.0.0.0         192.168.0.10              578      0x80000007     0xb041
-
  OSPFv3-Router             0.0.0.0         192.168.0.20              579      0x80000007     0x1085
-
  OSPFv3-Router             0.0.0.0         192.168.0.30              580      0x80000007      0xe89
-
  Inter-Area-Prefix         1.0.0.0         192.168.0.20              624      0x80000004     0xd4b7
15000
  Inter-Area-Prefix         1.0.0.0         192.168.0.30              624      0x80000004     0x2208
25000
  Inter-Area-Prefix         1.0.0.0         192.168.0.40              623      0x80000004     0x5c1c
15000
  Inter-Area-Prefix         1.0.0.0         192.168.0.50              623      0x80000004     0xf9ee
10000
  Intra-Area-Prefix         1.0.0.0         192.168.0.10              578      0x80000006     0x7af2
-
  Intra-Area-Prefix         1.0.0.0         192.168.0.20              579      0x80000005     0xf646
-
  Intra-Area-Prefix         1.0.0.0         192.168.0.30              580      0x80000006     0x56b2
-
  Link                      6.0.8.0         192.168.0.10              583      0x80000005     0x7d82
-
  Link                      6.0.8.0         192.168.0.20              584      0x80000005     0xa51a
-
  Link                      6.0.32.3        192.168.0.10              623      0x80000004     0x17d2
-
  Link                      6.0.32.3        192.168.0.20              624      0x80000004     0x3f6a
-
Instance: default, Address family: ipv6, Instance ID: 0
  Type                      Link State ID   Advertising Router        Age        Sequence   Checksum
Cost
  OSPFv3-External           1.0.0.0         192.168.0.60              619      0x80000004     0xa367
16777214
  OSPFv3-External           2.0.0.0         192.168.0.60              619      0x80000004     0x738a
16777214
<...>
```

Example 2: OSPF database detailed information

```
supervisor@rtbrick>SPINE01: cfg> show ospf database detail
Instance: default, Address family: ipv4, Area: 0.0.0.0 LSAs
  LSA ID: 192.168.0.10
    Advertising router: 192.168.0.10, LSA type: Router
    Sequence number: 0x80000008, Checksum: 0x262a, LSA age: 719s
    Length: 132, Options: *|-|-|-|-|-|E|*, Flags: -|-|-|-
    Number of links: 9
      Link ID: 12.0.0.2
        Link data: 12.0.0.1, Type: Transit
        Type of service: 0, Metric: 10000
      Link ID: 192.168.0.20
        Link data: 12.1.0.1, Type: P2P
        Type of service: 0, Metric: 20000
      Link ID: 12.1.0.0
        Link data: 255.255.255.252, Type: Stub
        Type of service: 0, Metric: 20000
      Link ID: 192.168.0.20
        Link data: 12.2.0.0, Type: P2P
        Type of service: 0, Metric: 40000
      Link ID: 12.2.0.0
```

```
          Link data: 255.255.255.254, Type: Stub
          Type of service: 0, Metric: 40000
        Link ID: 12.3.0.2
          Link data: 12.3.0.1, Type: Transit
          Type of service: 0, Metric: 30000
        Link ID: 11.0.0.5
          Link data: 11.0.0.1, Type: Transit
          Type of service: 0, Metric: 60000
        Link ID: 192.168.0.10
          Link data: 255.255.255.255, Type: Stub
          Type of service: 0, Metric: 10000
        Link ID: 192.168.0.11
          Link data: 255.255.255.255, Type: Stub
          Type of service: 0, Metric: 10000
    LSA ID: 192.168.0.20
      Advertising router: 192.168.0.20, LSA type: Router, Router ID: 192.168.0.20
      Sequence number: 0x80000009, Checksum: 0xddd7, LSA age: 720s
      Interface: ifl-0/0/0/100, Neighbor address: 12.1.0.2
      Length: 156, Options: *|-|-|-|-|-|E|*, Flags: -|-|-|B
      Number of links: 11
        Link ID: 12.0.0.2
          Link data: 12.0.0.2, Type: Transit
          Type of service: 0, Metric: 10000
        Link ID: 192.168.0.10
          Link data: 12.1.0.2, Type: P2P
          Type of service: 0, Metric: 20000
        Link ID: 12.1.0.0
          Link data: 255.255.255.252, Type: Stub
          Type of service: 0, Metric: 20000
<...>
```

Example 3: OSPF database for an advertising router

```
supervisor@rtbrick>SPINE01: cfg> show ospf database advertising-router 192.168.0.10
Instance: default, Address family: ipv4, Area: 0.0.0.0
  Type                    Link State ID   Advertising Router         Age        Sequence   Checksum
Cost
  Router                  192.168.0.10    192.168.0.10               791      0x80000008     0x262a
-
Instance: default, Address family: ipv6, Instance ID: 0, Area: 0.0.0.0
  Type                    Link State ID   Advertising Router         Age        Sequence   Checksum
Cost
  OSPFv3-Router           0.0.0.0         192.168.0.10               791      0x80000007     0xb041
-
  Intra-Area-Prefix       1.0.0.0         192.168.0.10               791      0x80000006     0x7af2
-
  Link                    6.0.8.0         192.168.0.10               796      0x80000005     0x7d82
-
  Link                    6.0.32.3        192.168.0.10               836      0x80000004     0x17d2
-
  Link                    6.8.8.0         192.168.0.10               836      0x80000004     0x677e
-
  Link                    6.8.32.3        192.168.0.10               836      0x80000004     0x2b9b
-
  Link                    6.32.8.0        192.168.0.10               795      0x80000006     0x3893
-
```

OSPF SPF Result

Displays SPF results.

Syntax:

show ospf spf result <options>

Option	Description
-	Without any option, the command displays the SPF result of all instances.
area <area-id>	Displays SPF result for the specified area.
instance <instance-name>	Name of the instance
node-id <node-id>	Displays SPF result for the specified node identifier.
area <area-id> <node-id>	Displays SPF result for the specified node identifier for a specified area.
instance <instance-name> area <area-id>	Displays SPF result for the specified area for a given instance.
instance <instance-name> node-id <node-id>	Displays SPF result for the specified node identifier for a given instance.
<afi>	Displays SPF result information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. You can display SPF result information for the specified address family using filters such as interface name, detail, and instance.

Example 1: OSPF SPF Result for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf spf result
Instance: default, Address family: ipv4, Area: 0.0.0.0
  Node ID         Type            Cost     Advertising Router  Flags        Neighbor Node         Interface
Nexthop
  12.0.0.2        network         10000    192.168.0.20        -|-|-|-      -                     local
-
  12.3.0.2        network         30000    192.168.0.20        -|-|-|-      -                     local
-
  23.0.0.3        network         20000    192.168.0.30        -|-|-|-      192.168.0.20          ifl-0/0/0/1
12.0.0.2
  11.0.0.5        network         55000    192.168.0.50        -|-|-|-      192.168.0.20          ifl-0/0/0/1
12.0.0.2
  192.168.0.10    router          0        192.168.0.10        -|-|-|-      -                     local
-
  192.168.0.20    router          10000    192.168.0.20        -|-|-|B      192.168.0.20          ifl-0/0/0/1
12.0.0.2
  192.168.0.30    router          20000    192.168.0.30        -|-|-|B      192.168.0.20          ifl-0/0/0/1
12.0.0.2
  192.168.0.40    router          55000    192.168.0.40        -|-|-|B      192.168.0.20          ifl-0/0/0/1
12.0.0.2
  192.168.0.50    router          55000    192.168.0.50        -|-|-|B      192.168.0.20          ifl-0/0/0/1
12.0.0.2
Instance: default, Address family: ipv6, Area: 0.0.0.0, Instance ID: 0
  Node ID         Type            Cost     Advertising Router  Flags        Neighbor Node         Interface
Nexthop
```

```
  192.168.0.10    router          0        192.168.0.10        -|-|-|-      -                     local
-
  6.0.8.0         network         10000    192.168.0.20        -|-|-|-      -                     local
-
  6.8.32.3        network         30000    192.168.0.20        -|-|-|-      -                     local
-
  192.168.0.20    router          10000    192.168.0.20        -|-|-|B      192.168.0.20          ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  6.0.8.0         network         20000    192.168.0.30        -|-|-|-      192.168.0.20          ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  192.168.0.30    router          20000    192.168.0.30        -|-|-|B      192.168.0.20          ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  192.168.0.40    router          55000    192.168.0.40        -|-|-|B      192.168.0.20          ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  6.48.8.0        network         55000    192.168.0.50        -|-|-|-      192.168.0.20          ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  192.168.0.50    router          55000    192.168.0.50        -|-|-|B      192.168.0.20          ifl-0/0/0/1
fe80::7845:9aff:fec0:0
```

Example 2: OSPF SPF Result for the specified node identifier for the given area

```
supervisor@rtbrick>SPINE01: op> show ospf spf result area 0.0.0.0 node-id 192.168.0.10
Instance: default, Address family: ipv4, Area: 0.0.0.0
  Node ID         Type            Cost     Advertising Router  Flags        Neighbor Node         Interface
Nexthop
  192.168.0.10    router          0        192.168.0.10        -|-|-|-      -                     local
-
Instance: default, Address family: ipv6, Area: 0.0.0.0, Instance ID: 0
  Node ID         Type            Cost     Advertising Router  Flags        Neighbor Node         Interface
Nexthop
  192.168.0.10    router          0        192.168.0.10        -|-|-|-      -                     local
-
```

OSPF SPF Log

Displays SPF Log information.

Syntax:

show ospf spf log <options>

Option	Description
-	Without any option, the command displays the SPF log of all instances.
instance <instance-name>	Displays SPF log for the specified instance.
<afi>	Displays SPF log information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. You can display SPF log information for the specified address family using filters such as interface name, detail, and instance.

Example 1: OSPF SPF Result for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf spf log
Instance: default
  Router ID: 192.168.0.10
    Schedule timestamp: 2024-04-30 11:19:51, Area ID: 0.0.0.0, LSA type: Router
    Reason: Router LSA change, Back off timer: 50, LSA count: 1
    LS ID: 0.0.0.0, Number of schedule request: 1
    SPF start time: 2024-04-30 11:19:51, Number of nodes: 1, Number of links: 0
    Number of stub links: 0, SPF init time: 31us, SPF run time: 394us
    Router LSA change count: 1, Network LSA change count: -
    Prefix changes: -, Sequence number: 1
  Router ID: 192.168.0.10
    Schedule timestamp: 2024-04-30 11:19:51, Area ID: 0.0.0.0, LSA type: Router
    Reason: Router LSA change, Back off timer: 50, LSA count: 1
    LS ID: 192.168.0.10, Number of schedule request: 1
    SPF start time: 2024-04-30 11:19:51, Number of nodes: 1, Number of links: 0
    Number of stub links: 7, SPF init time: 7us, SPF run time: 311us
    Router LSA change count: 1, Network LSA change count: -
    Prefix changes: -, Sequence number: 2
  Router ID: 192.168.0.10
    Schedule timestamp: 2024-04-30 11:19:51, Area ID: 0.0.0.0, LSA type: Link
    Reason: Unknown map - (value) 0xa, Back off timer: 200, LSA count: 5
    LS ID: 6.0.8.0, Number of schedule request: 5
    SPF start time: 2024-04-30 11:19:51, Number of nodes: 1, Number of links: 0
    Number of stub links: 0, SPF init time: 7us, SPF run time: 172us
    Router LSA change count: 1, Network LSA change count: -
    Prefix changes: -, Sequence number: 3
  Router ID: 192.168.0.20
    Schedule timestamp: 2024-04-30 11:19:51, Area ID: 0.0.0.0, LSA type: Router
    Reason: Router LSA change, Back off timer: 5000, LSA count: 6
    LS ID: 192.168.0.20, Number of schedule request: 6
    SPF start time: 2024-04-30 11:19:56, Number of nodes: 3, Number of links: 6
    Number of stub links: 20, SPF init time: 35us, SPF run time: 182us
    Router LSA change count: 7, Network LSA change count: -
    Prefix changes: -, Sequence number: 4
<...>
```

OSPF Route

Displays OSPF routing table information.

Syntax:

show ospf route <options>

Option	Description
-	Without any option, the command displays the OSPF route information for all instances.
area <area-id>	OSPF route information for the given area.
instance <instance-name>	OSPF route information for the given instance.

Option	Description
instance <instance-name> <afi>	Displays OSPF route information for the specified address family and instance. Supported AFI values are 'ipv4' and 'ipv6'.
instance <instance-name> <afi> <safi>	Displays OSPF route information for the specified address family (AFI), sub-address family (SAFI), and instance. Supported AFI values are 'ipv4' and 'ipv6'. Supported SAFI values are 'unicast', and 'labeled-unicast''.
instance <instance-name> label <label>	Displays OSPF route information for the specified label and instance.
instance <instance-name> mpls unicast label <label> | type <type>	Displays OSPF route information for the specified MPLS unicast label or type for the instance.
prefix <ip>	Displays OSPF route information for the specified match prefix.
type	Displays information for OSPF route type. The route types include external-type-1, external-type-2, inter-area, intra-area, and ospf-direct.
<afi> <safi>	Displays OSPF route information for the specified address family (AFI) and sub-address family (SAFI). Supported AFI values are 'ipv4' and 'ipv6'. Supported SAFI values are 'unicast', and 'labeled-unicast''
label <label>	Displays information about the OSPF-labeled routes.
mpls unicast <label | type>	Displays information about OSPF MPLS routes.
area-border	Displays the OSPF Area Border Router (ABR) information. Refer to section "3.1.7. OSPF Route ABR" for the interface configuration details.
autonomous-system-boundary	Displays Autonomous System Border Router information. Refer to section "3.1.7. OSPF Route ABR" for the interface configuration details.

Example: OSPF route information for the default instance

```
supervisor@rtbrick>SPINE01: op> show ospf route
Instance: default, AFI: ipv4, SAFI: unicast
  Prefix                     Area             Type             Cost      Next Hop        Interface
  11.0.0.0/24                0.0.0.0          intra-area       55000     12.0.0.2        ifl-0/0/0/1
  12.0.0.0/23                0.0.0.0          ospf-direct      10000     n/a             local
  12.1.0.0/30                0.0.0.0          ospf-direct      20000     n/a             local
  12.2.0.0/31                0.0.0.0          ospf-direct      40000     n/a             local
  12.3.0.0/17                0.0.0.0          ospf-direct      30000     n/a             local
  23.0.0.0/24                0.0.0.0          intra-area       20000     12.0.0.2        ifl-0/0/0/1
  23.1.0.0/24                0.0.0.0          intra-area       25000     12.0.0.2        ifl-0/0/0/1
  24.0.1.0/24                0.0.0.0          inter-area       25000     12.0.0.2        ifl-0/0/0/1
  24.1.1.0/24                0.0.0.0          inter-area       20000     12.0.0.2        ifl-0/0/0/1
  25.0.1.0/24                0.0.0.0          inter-area       35000     12.0.0.2        ifl-0/0/0/1
  192.168.0.10/32            0.0.0.0          ospf-direct      10000     n/a             local
  192.168.0.11/32            0.0.0.0          ospf-direct      10000     n/a             local
  192.168.0.20/32            0.0.0.0          intra-area       20000     12.0.0.2        ifl-0/0/0/1
  192.168.0.21/32            0.0.0.0          intra-area       20000     12.0.0.2        ifl-0/0/0/1
  200.0.1.60/32                               external-type-1  16777215  12.0.0.2        ifl-0/0/0/1
  200.0.1.61/32                               external-type-1  16777215  12.0.0.2        ifl-0/0/0/1
  200.0.1.0/24                                external-type-1  16777215  12.0.0.2        ifl-0/0/0/1
  200.0.2.60/32                               external-type-1  16777215  12.0.0.2        ifl-0/0/0/1
  200.0.2.61/32                               external-type-1  16777215  12.0.0.2        ifl-0/0/0/1
  200.0.2.0/24                                external-type-1  16777215  12.0.0.2        ifl-0/0/0/1
Instance: default, AFI: ipv6, SAFI: unicast, Instance ID: 0
  Prefix                                       Area             Type             Cost      Next Hop
               Interface
  11::/64                                      0.0.0.0          intra-area       55000
fe80::7845:9aff:fec0:0                      ifl-0/0/0/1
  12::/64                                      0.0.0.0          ospf-direct      10000     n/a
               local
  12:1::/64                                    0.0.0.0          ospf-direct      20000     n/a
               local
  12:2::/64                                    0.0.0.0          ospf-direct      40000     n/a
               local
  23::/64                                      0.0.0.0          intra-area       20000
fe80::7845:9aff:fec0:0                      ifl-0/0/0/1
  24:0:1::/64                                  0.0.0.0          inter-area       25000
fe80::7845:9aff:fec0:0                      ifl-0/0/0/1
  24:1:1::/64                                  0.0.0.0          inter-area       20000
fe80::7845:9aff:fec0:0                      ifl-0/0/0/1
  25:0:1::/64                                  0.0.0.0          inter-area       35000
fe80::7845:9aff:fec0:0                      ifl-0/0/0/1
  25:1:1::/64                                  0.0.0.0          inter-area       40000
fe80::7845:9aff:fec0:0                      ifl-0/0/0/1
<...>
```

OSPF Route Area Border

Displays the OSPF Area Border Router (ABR) information.

Syntax:

show ospf route area-border

Example: OSPF Route ABR information

```
supervisor@rtbrick>SPINE01: cfg> show ospf route area-border
  192.168.0.20    10000    192.168.0.20        -|-|-|B      ifl-0/0/0/1
12.0.0.2
  192.168.0.30    20000    192.168.0.30        -|-|-|B      ifl-0/0/0/1
12.0.0.2
  192.168.0.40    55000    192.168.0.40        -|-|-|B      ifl-0/0/0/1
12.0.0.2
  192.168.0.50    55000    192.168.0.50        -|-|-|B      ifl-0/0/0/1
```

```
12.0.0.2
  192.168.0.20    10000    192.168.0.20        -|-|-|B      ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  192.168.0.30    20000    192.168.0.30        -|-|-|B      ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  192.168.0.40    55000    192.168.0.40        -|-|-|B      ifl-0/0/0/1
fe80::7845:9aff:fec0:0
  192.168.0.50    55000    192.168.0.50        -|-|-|B      ifl-0/0/0/1
fe80::7845:9aff:fec0:0
```

OSPF Route Autonomous System Boundary

Displays Autonomous System Boundary Router information.

Syntax:

show ospf route autonomous-system-boundary

Example: OSPF Route ASBR information

```
supervisor@rtbrick>SPINE01: cfg> show ospf route autonomous-system-boundary
Instance: default, Address family: ipv4
  Node ID         Cost     Advertising Router  Flags        Interface
Nexthop
  192.168.0.60    40000    192.168.0.20                     ifl-0/0/0/1
12.0.0.2
Instance: default, Address family: ipv6, Instance ID: 0
  Node ID         Cost     Advertising Router  Flags        Interface
Nexthop
  192.168.0.60    40000    192.168.0.20                     ifl-0/0/0/1
fe80::7845:9aff:fec0:0
```

OSPF LSA Request List

Displays the list of all link-state advertisements (LSAs) requests that have been sent or received by a router.

Syntax:

show ospf request-list <options>

Option	Description
-	Without any option, this command displays the list of all link-state advertisement (LSA) requests that have been sent from the router.

Option	Description
detail	Provides detailed information on the requests that have been sent from the router.
area <area-id>	OSPF request-list information for the given area.
instance <instance-name>	OSPF request-list information for the given instance.
<afi>	Displays request-list information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'.

Example 1: OSPF LSA requests sent to a neighbor

```
supervisor@rtbrick>SPINE01: op> show ospf request-list
Instance: default
  Type            Link State ID   Advertising Router          Age        Sequence   Checksum
  Summary-Network 11.0.0.0        198.51.100.20                 42      0x80000003     0x76e5
  Summary-Network 12.0.0.0        198.51.100.20                 42      0x80000003     0x603d
  Summary-Network 12.1.0.0        198.51.100.20                 42      0x80000003     0x481f
  Summary-Network 12.2.0.0        198.51.100.20                 42      0x80000003     0x4aab
  Summary-Network 12.3.0.0        198.51.100.20                 42      0x80000003     0xc5e4
  Summary-Network 23.0.0.0        198.51.100.20                 42      0x80000003     0xd5bb
  Summary-Network 23.1.0.0        198.51.100.20                 42      0x80000003     0xca2a
```

Example 2: Detailed information for OSPF LSA requests sent to a neighbor

```
supervisor@rtbrick>SPINE01: op> show ospf request-list detail
Instance: default LSAs
  LSA ID: 11.0.0.0
    Advertising router: 198.51.100.20, LSA type: Summary-Network, Router ID: 192.168.0.20
    Sequence number: 0x80000003, Checksum: 0x76e5, LSA age: 42
    Interface: ifl-0/0/0/1, Neighbor address: 25.0.1.2
    Length: 0, Options: *|-|-|-|-|-|-|*
  LSA ID: 12.0.0.0
    Advertised router: 198.51.100.20, LSA type: Summary-Network, Router ID: 192.168.0.20
    Sequence number: 0x80000003, Checksum: 0x603d, LSA age: 42
    Interface: ifl-0/0/0/1, Neighbor address: 25.0.1.2
    Length: 0, Options: *|-|-|-|-|-|-|*
  LSA ID: 12.1.0.0
    Advertised router: 198.51.100.20, LSA type: Summary-Network, Router ID: 192.168.0.20
    Sequence number: 0x80000003, Checksum: 0x481f, LSA age: 42
    Interface: ifl-0/0/0/1, Neighbor address: 25.0.1.2
    Length: 0, Options: *|-|-|-|-|-|-|*
  LSA ID: 12.2.0.0
    Advertised router: 198.51.100.20, LSA type: Summary-Network, Router ID: 192.168.0.20
    Sequence number: 0x80000003, Checksum: 0x4aab, LSA age: 42
    Interface: ifl-0/0/0/1, Neighbor address: 25.0.1.2
    Length: 0, Options: *|-|-|-|-|-|-|*

 <...>
```

OSPF Transmission List

Displays the list of all LSAs waiting to be re-sent or transmitted from the router.

Syntax:

show ospf transmit-list <option>

Option	Description
-	Without any option, this command displays the transmit list of all link-state advertisements (LSA).
area <area-id>	OSPF transmit-list information for the given area.
instance <instance-name>	OSPF transmit-list information for the given instance.
<afi>	Displays transmit-list information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'.

Example: OSPF LSA requests waiting to be transmitted.

```
supervisor@rtbrick>SPINE01: op> show ospf transmit-list
Instance: default, Area: 0.0.0.1, Interface: ifl-0/0/4/1, Neighbor: 25.0.1.5
  LSA ID                  LS type                 Advertising router        Transmit interval
Retransmit count
  11.0.0.0                Summary-Network         198.51.100.20              5000                    1
  12.0.0.0                Summary-Network         198.51.100.20              5000                    1
  23.0.0.0                Summary-Network         198.51.100.20              5000                    1
  12.1.0.0                Summary-Network         198.51.100.20              5000                    1
  23.1.0.0                Summary-Network         198.51.100.20              5000                    1
  12.2.0.0                Summary-Network         198.51.100.20              5000                    1
  12.3.0.0                Summary-Network         198.51.100.20              5000                    1
```

OSPF Statistics

Displays OSPF statistics information.

Syntax:

show ospf statistics <options>

Option	Description
interface <interface-name>	Displays packet statistics information for the specified interface.

Option	Description
interface <interface-name> detail	Displays detailed packet statistics information for the specified interface.
neighbor <Neighbor-address>	Displays packet statistics information for the specified neighbor.
neighbor <Neighbor-address> <detail>	Displays detailed packet statistics information for the specified neighbor.
<afi>	Displays packet statistics information for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'.

```
supervisor@rtbrick>SPINE01: op> show ospf statistics interface ifl-0/0/0/1 detail
Instance: default, Address family: ipv4
  Interface: ifl-0/0/0/1, Peer address: 12.0.0.2
    Hello packet:
      Received packets: 658, Sent packets: 659, Total errors: 0, Unsupported option: 0
      Area mismatch: 0, Area type option mismatch: 0, Dead interval mismatch: 0
      Hello interval mismatch: 0, Mask mismatch: 0, Self router ID: 0
      Obj add fail: 0, Source address mismatch: , Misc: 0
    DD packet:
      Received packets: 3, Sent packets: 4, Total errors: 1, Unsupported option: 0
      Invalid state packet rcvd: 0, MTU mismatch: 0, DD obj add fail: 0, Misc: 0, Negotiation fail: 0
      Master bit mismatch: 0, Exchange state init pkt: 0, Capabilities mismatch: 0
      Expected seq mismatch: 0, Full state init pkt: 0, Invalid lsa: 0, Invalid external lsa: 0,
      Lsdb Absent: 0, Lsa lookup fail: 0, Ls req add fail: 0
    LS request packet:
      Received packets: 0, Sent packets: 0, Total errors: 0, Invalid LSA type: 0
      Invalid state packet rcvd: 0, LSA lookup error: 0, LSA lookup fail: 0
      LSA obj add fail: 0, Misc: 0
    LS update packet:
      Received packets: 0, Sent packets: 0, Total errors: 0, Invalid LSA type: 0
      Zero length LSA: 0, LSA length exceeded: 0, LSA checksum fail: 0
      Invalid state packet rcvd: 0, LSA obj add fail: 0, Invalid LSA payload: 0, Misc: 0
    Ls ack packet:
      Received packets: 0, Sent packets: 0, Total errors: 0, LSA obj add fail: 0
      Invalid state packet rcvd: 0, Misc: 0
    Sanity errors:
      Payload max len error: 0, Payload min len error: 0, Invalid version: 0
      Invalid auth data len: 0, Auth data missing: 0, Invalid packet min len: 0
      Invalid area ID: 0, Invalid network mask: 0, Authentication fail: 1
Instance: default, Address family: ipv6, Instance ID: 0
  Interface: ifl-0/0/0/1, Peer address: fe80::7845:9aff:fec0:0
    Hello packet:
      Received packets: 658, Sent packets: 658, Total errors: 0, Unsupported option: 0
      Area mismatch: 0, Area type option mismatch: 0, Dead interval mismatch: 0
      Hello interval mismatch: 0, Mask mismatch: 0, Self router ID: 0
      Obj add fail: 0, Source address mismatch: , Misc: 0
    DD packet:
      Received packets: 4, Sent packets: 4, Total errors: 1, Unsupported option: 0
      Invalid state packet rcvd: 0, MTU mismatch: 0, DD obj add fail: 0, Misc: 0, Negotiation fail: 0
      Master bit mismatch: 0, Exchange state init pkt: 0, Capabilities mismatch: 0
      Expected seq mismatch: 0, Full state init pkt: 0, Invalid lsa: 0, Invalid external lsa: 0,
      Lsdb Absent: 0, Lsa lookup fail: 0, Ls req add fail: 0
    LS request packet:
      Received packets: 1, Sent packets: 1, Total errors: 0, Invalid LSA type: 0
      Invalid state packet rcvd: 0, LSA lookup error: 0, LSA lookup fail: 0
      LSA obj add fail: 0, Misc: 0
    LS update packet:
      Received packets: 4, Sent packets: 5, Total errors: 0, Invalid LSA type: 0
```

```
        Zero length LSA: 0, LSA length exceeded: 0, LSA checksum fail: 0
        Invalid state packet rcvd: 0, LSA obj add fail: 0, Invalid LSA payload: 0, Misc: 0
      Ls ack packet:
        Received packets: 4, Sent packets: 4, Total errors: 0, LSA obj add fail: 0
        Invalid state packet rcvd: 0, Misc: 0
      Sanity errors:
        Payload max len error: 0, Payload min len error: 0, Invalid version: 0
        Invalid auth data len: 0, Auth data missing: 0, Invalid packet min len: 0
        Invalid area ID: 0, Invalid network mask: 0, Authentication fail: 1
```

OSPF Clear Commands

Clear OSPF Neighbor

Clear OSPF neighbor state information.

Syntax:

clear ospf neighbor <options>

Option	Description
-	Without any option, the command clears all the OSPF neighbors.
instance <instance-name>	Clears OSPF neighbor information for the specified instance.
instance <instance-name> <afi>	Clears OSPF neighbor for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. Supported AFI values are 'ipv4' and 'ipv6'. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
instance <instance-name> <afi> area <area-id>	Clears OSPF neighbor for the specified area of the specified instance and address family.
instance <instance-name> <afi> area <area-id> interface <interface-name>	Clears OSPF neighbor for the specified interface for the specified area and address family of the specified instance.
force	Forcefully clears all the OSPF neighbors. This may impact DR/BDR election.
force instance <instance-name>	Forcefully clears the neighbor for the specified instance.

Option	Description
force instance <instance-name> <afi>	Forcefully clears OSPF neighbor for the specified address family.
force instance <instance-name> <afi> area <area-id>	Forcefully clears OSPF neighbor for the specified area of the specified instance and address family.
force instance <instance-name> <afi> area <area-id> interface <interface-name>	Forcefully clears OSPF neighbor for a specific interface and area of the specified instance and address family.

Example:

```
supervisor@rtbrick>SPINE01: cfg> clear ospf neighbor instance default ipv6 instance-id 0 area 0 interface
ifl-0/0/0/1
Triggered clear neighbor successfully
```

To access the Operational State API that corresponds to this CLI, click here.

Clear OSPF Statistics

Clear the OSPF statistics for all instances or a specified instance.

Syntax:

clear ospf statistics <options>

Option	Description
-	Without any option, the command clears all the OSPF statistics.
instance <instance-name>	Clears OSPF statistics information for the specified instance.

Option	Description
instance <instance-name> <afi>	Clears OSPF statistics for the specified address family of the specified instance. Supported AFI values are 'ipv4' and 'ipv6'. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
instance <instance-name> <afi> area <area-id>	Clears OSPF statistics for the specified area of the specified instance and address family.
instance <instance-name> <afi> area <area-id> interface <interface-name>	Clears OSPF statistics for the specified interface for the specified area of the specified instance and address family.

Example:

```
supervisor@rtbrick>SPINE01: cfg> clear ospf statistics instance default ipv6 instance-id 0 area 0.0.0.0
interface ifl-0/0/0/1
Cleared statistics for all ipv6 neighbors under Instance [default] Area [0.0.0.0] Interface [ifl-0/0/0/1]
```

Clear OSPF Database

Clear the OSPF database for all instances.

Syntax:

clear ospf database <options>

Option	Description
-	Without any option, the command clears all the OSPF database information.
<afi>	Clears OSPF database for the specified address family. Supported AFI values are 'ipv4' and 'ipv6'. Supported AFI values are 'ipv4' and 'ipv6'. When using an IPv6 address family with OSPFv3, an instance ID ranging from 0 to 31 must be specified.
<afi> instance <instance-name>	Clears OSPF database for the specified area of the specified instance and address family.

Example:

```
supervisor@rtbrick>SPINE01: cfg> clear ospf database ipv6 instance default instance-id 0
Triggered clear database successfully
```

2.8. LDP

2.8.1. LDP Overview

Label distribution protocol (LDP) is the most commonly used protocol in the MPLS network. It generates and distributes labels and thus helps in MPLS packet switching and forwarding. By using LDP, label-switching routers in an MPLS network can exchange label mapping information to create label-switched paths (LSPs) for switching data packets. RtBrick FullStack (RBFS) supports Dual-stack, which means LDP can exchange FEC-label bindings over either IPv4 or IPv6 networks.

Peer Discovery

LDP sends UDP multicast hello packets to discover its neighbors and establishes neighbor adjacency with other directly connected label switch routers (LSRs). The hello message is periodically sent on LDP-enabled interfaces.

Session Establishment

After peer discovery, “initialization messages” are sent to each other. In these messages, the session Parameters are sent to each other. The LDP sessions are maintained by periodic keep-alive message.

After the LDP neighbors are discovered, the TCP session is established and the LDP FSM is triggered, and LDP session becomes operational. LSRs start exchanging label mapping information with each other.

Dual-stack LDP

By default, RBFS is dual-stack capable, which means it can exchange IPv4/IPv6 FEC bindings over IPv4/IPv6 media (LDP over IPv4/IPv6).

To enable or disable a particular address family in RBFS, use "status <enable|disable>" CLI. For details, see LDP Address Family Configuration.

When LDP is enabled on an interface that supports both IPv4 and IPv6, LDP will start exchanging IPv4 hellos. To send an IPv6 hello, the source IPv6 address must be configured. For details about configuring the source address, see LDP Instance Configuration.

By default, both IPv4 and IPv6 hello will use the same transport preference as IPv6, but this can be changed by using the "connection-preference <ipv4|ipv6>" CLI. For details, see LDP Instance Configuration.

The following points should be noted regarding this functionality:

Source address:

- The source address specifies the transport address, either IPv4 or IPv6, to be used by the LDP protocol.
- When the source address is not configured, only an IPv4 transport connection will be established. As a result, only IPv4 FECs will be installed. To install IPv6 FECs, the source IPv6 address needs to be configured.

Connection Preference:

- When LDP is configured for dual-stack operation, the transport connection preference is IPv6 by default, and LDP will establish TCP connections over IPv6. To establish an IPv4 LDP session, the connection preference must be set to IPv4, and LDP will distribute IPv4 and IPv6 FECs over IPv4 connections.

Targeted Label Distribution Protocol (T-LDP)

RBFS supports Targeted Label Distribution Protocol (T-LDP) that enables a router to establish an LDP session with a router that is not directly connected. T-LDP is commonly used in MPLS networks to facilitate the distribution of labels across various segments of the network.

Label Generation

LDP generates label bindings for the IP addresses of the LDP-enabled loopback interfaces and then advertises them to all neighbors.

Label Management Modes

Label Advertisement Mode

LDP supports the Downstream Unsolicited feature in RBFS, where label bindings are advertised to all upstream neighbors. By default, label advertisement operates in the Downstream Unsolicited mode.

Label Distribution Control Mode

LDP supports the Ordered Label Distribution Control, where an LSR will initiate the transmission of the label mapping only for the prefix for which it has a label mapping from the next hop of the prefix or for which it is an egress.

Label Retention Mode

LDP supports the Liberal Label Retention Mode where all the label mapping advertisements for all routes received from all the LDP neighbors are retained.

Supported LDP Standards

RFC Number	Description
RFC 5036	LDP Specification The following modes are supported by RBFS for the features listed in RFC 5036: • Label advertisement: Downstream Unsolicited mode is supported but not Downstream on Demand mode. • Label distribution control: Ordered mode is supported, but not Independent mode. • Label retention: Liberal mode is supported, but not Conservative mode.
RFC 5283	LDP Extension for Inter-Area Label Switched Paths (LSPs)
RFC 5443	LDP IGP Synchronization
RFC 7552	IPv6 Dual-Stack

RFC and draft compliance are partial except as specified.

Supported LDP Features

The following LDP features are supported in this release of RBFS:

- Support for the following label management modes.

  Downstream unsolicited mode in label advertisement

  Ordered mode in the label distribution control

  Liberal mode in label retention
- Loop detection
- Targeted Label Distribution Protocol (T-LDP)
- Inter-area support
- Tracking IGP metric
- IGP LDP synchronization
- LDP Dual-stack support
- LDP TCP authentication
- LDP redistribution
- LDP policy configuration

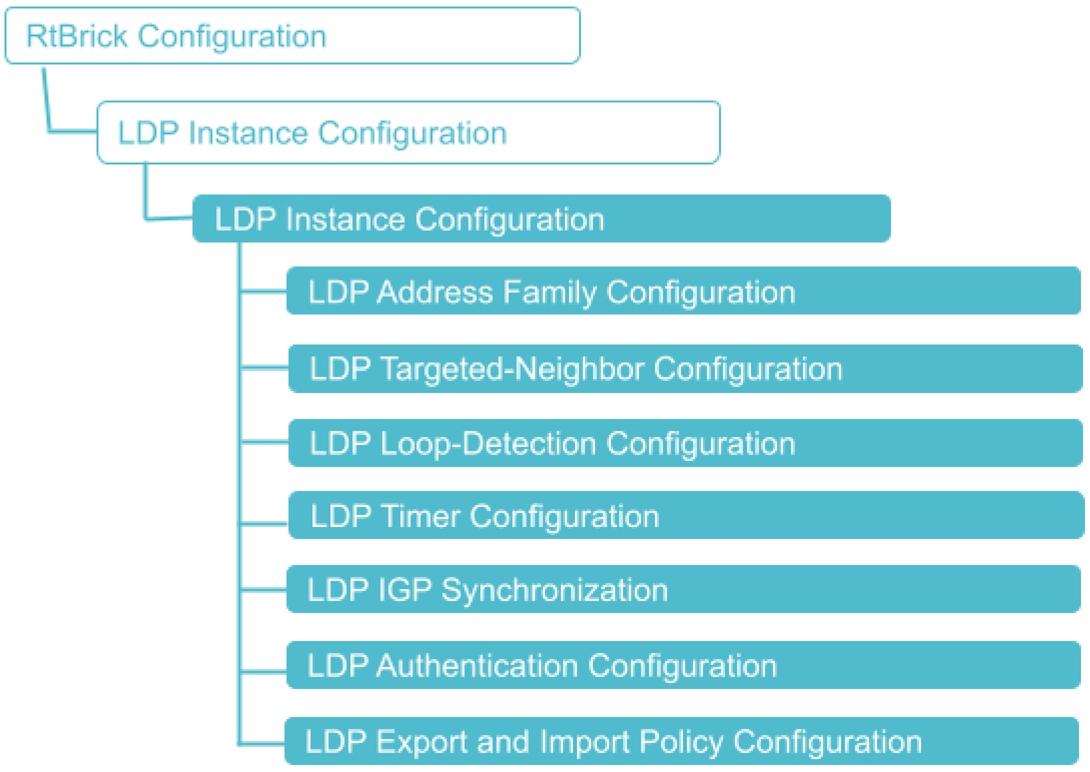
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

2.8.2. LDP Configuration

LDP Configuration Hierarchy

The diagram below illustrates the LDP configuration hierarchy.


RtBrick Configuration
LDP Instance Configuration
LDP Instance Configuration
LDP Address Family Configuration
LDP Targeted-Neighbor Configuration
LDP Loop-Detection Configuration
LDP Timer Configuration
LDP IGP Synchronization
LDP Authentication Configuration
LDP Export and Import Policy Configuration

Configuration Syntax and Commands

The following sections describe the LDP configuration syntax and commands.

LDP Instance Configuration

At this configuration hierarchy, you configure LDP protocol parameters which are generic to the LDP instance.

Syntax:

set instance <instance-name> **protocol ldp** <attribute> <value>

Attribute	Description
<instance-name>	Name of the LDP instance.
interface <name>	Name of the logical interface.
router-id <router-id>	Router identifier in IPv4 format.
address-family <afi>	Address family identifier (AFI). Supported values: ipv4, or ipv6. Refer to section LDP Address Family Configuration for LDP address family configuration details.

Attribute	Description
connection-preference <afi>	Specifies the connection preference for the TCP session. Supported values: ipv4, or ipv6. By default, IPv6 is used as the preferred TCP connection if an IPv6 source address is configured. Refer to section Dual-stack LDP for more information on the LDP Dual-stack behaviour.
igp-synchronization <…>	LDP IGP synchronization configuration. This option is supported only on interfaces running Intermediate System-to-System (IS-IS) or OSPFv2 processes. Refer to section LDP IGP Synchronization for LDP-IGP synchronization configuration details.
source-address <ipv4|ipv6> <source-address>	Use the specified IP addresses (IPv4 or IPv6) as the transport address for the LDP session. For LDP over IPv6, the IPv6 source address is mandatory. Refer to section Dual-stack LDP for more information on the LDP Dual-stack behavior.
loop-detection <…>	The LDP loop detection feature enables LDP to detect loops during an LSP establishment. Refer to section LDP Loop Detection Configuration for the loop detection configuration details.
timer <…>	Specifies the Hello hold time, Hello interval, Keepalive hold time, and Keepalive interval. Refer to section LDP Timer Configuration for the LDP timer configuration details.
session <ipv4|ipv6> <address> authentication-id <…>	Specifies an IPv4 or IPv6 LDP session attributes to apply TCP authentication. Refer to section LDP Authentication Configuration.
targeted-neighbor <ipv4|ipv6> <neighbor-address> <source-address>	Specifies the IPv4 or IPv6 address of the targeted LDP neighbor. Refer to section LDP Targeted Neighbor Configuration.
session <ipv4|ipv6> <address> export-policy|import-policy <…>	Specifies an IPv4 or IPv6 LDP session attributes to apply import/export policy configurations. Refer to section LDP Import and Export Policy Configuration.

Example: LDP Instance Configuration

The following example shows some LDP instance configuration attributes. The further LDP configurations like timers and loop detection are shown in the examples in the subsequent sections.

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp
{
    "rtbrick-config:ldp": {
      "router-id": "198.51.100.1",
      "interface": [
        {
          "name": "ifl-0/0/0/1"
        },
        {
          "name": "ifl-0/0/0/100"
        },
        {
          "name": "ifl-0/0/0/101"
        },
        {
          "name": "ifl-0/0/1/102"
        },
        {
          "name": "ifl-0/0/2/1"
        },
        {
          "name": "ifl-0/0/3/1"
        },
        {
          "name": "lo-0/0/0/1"
        },
        {
          "name": "lo-0/0/0/2"
        },
        {
          "name": "lo-0/0/0/3"
        },
        {
          "name": "lo-0/0/0/4"
        },
        {
          "name": "lo-0/0/0/5"
        }
      ]
    }
  }
supervisor@rtbrick>SPINE01: cfg>
```

LDP Targeted Neighbor Configuration

The targeted neighbor configuration establishes an LDP neighbor with a remote neighbor that is not directly connected.

Syntax:

set instance <instance> **protocol ldp targeted-neighbor** <attribute> <value>

Attribute	Description
ipv4 <neighbor-address>	IPv4 address of the remote LDP neighbor.
ipv4 <neighbor-address> <source-address>	Neighbhor address to which ldp hello is sent
ipv4 <neighbor-address> <source-address> hold-time <hold-time>	Targeted hold time, <0-65535>, default 45s
ipv4 <neighbor-address> <source-address> interval <interval>	Targeted hello messages interval, <0-65535>, default 15s
ipv4 <neighbor-address> <source-address> neighbor-type <receiver |sender>	Specifies the neighbor type, that is, sender or receiver.
ipv6 <neighbor-address>	IPv6 address of the remote LDP neighbor.
ipv6 <neighbor-address> <source-address>	Neighbhor address to which ldp hello is sent
ipv6 <neighbor-address> <source-address> hold-time <hold-time>	Targeted hold time, <0-65535>, default 45s
ipv6 <neighbor-address> <source-address> interval <interval>	Targeted hello messages interval, <0-65535>, default 15s
ipv6 <neighbor-address> <source-address> neighbor-type <receiver |sender>	Specifies the neighbor type, that is, sender or receiver.

Example 1: LDP Targeted Neighbor Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp
targeted-neighbor
{
  "rtbrick-config:targeted-neighbor": {
    "ipv4": [
      {
        "neighbor-address": "198.51.100.1",
        "source-address": "198.51.100.2"
      },
      {
        "neighbor-address": "198.51.100.5",
        "source-address": "198.51.100.2"
      }
    ],
    "ipv6": [
      {
        "neighbor-address": "2001:DB8::1",
        "source-address": "2001:DB8::2"
      },
      {
        "neighbor-address": "2001:DB8::5",
        "source-address": "2001:DB8::2"
      }
    ]
  }
}
supervisor@rtbrick>SPINE01: cfg>
```

LDP Address Family Configuration

The address-family command allows you to enable the address families that LDP will route and configure settings that are specific to that address family.

Syntax:

set instance <instance-name> **protocol ldp address-family** <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6
<afi> status <enable|disable>	Enable or disable address family. By default, both IPv4 and IPv6 address families are enabled, as LDP supports dual stack. Refer to section Dual-stack LDP for more information on the LDP Dual-stack behavior.
<afi> redistribute <source>	Specifies the source from which the routes are to be redistributed. The available options include direct, ipoe, isis, ospf, ppp, and static.

Attribute	Description
<afi> redistribute <source> policy <policy>	Specifies the name of the policy map. The redistribute attach point allows routes from other sources to be advertised by LDP. The policy can be applied only to the routes that are redistributed from other sources to LDP.

Example 1: LDP Address Family Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp
address-family
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv6",
      "status": "disable"
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

Example 2: LDP Redistribution Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp
address-family ipv4 redistribute direct
{
  "rtbrick-config:redistribute": [
    {
      "source": "direct"
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

Example 3: LDP Policy Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp
address-family ipv4 redistribute
{
  "rtbrick-config:redistribute": [
    {
      "source": "direct",
      "policy": "filter-link-addres"
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

LDP Loop Detection Configuration

The LDP loop detection feature enables LDP to detect loops during an LSP establishment.

Syntax:

set instance <instance-name> **protocol ldp loop-detection** <attribute> <value>

Attribute	Description
hop-count <hop-count>	Specifies the hop count limit for loop detection. Range: 0-255. Default: 32.
status <enable|disable>	Enables or disables loop detection. By default, this option is disabled. When this option is enabled, both hop count and path vector are enabled.
vector-length <vector-length>	Specifies the path vector length limit for loop detection. Range: 0-255. Default: 32.

Example 1: LDP Loop Detection Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp loop-
detection
{
    "rtbrick-config:loop-detection": {
      "enable": "true",
      "hop-count": 64,
      "vector-length": 64
    }
  }
supervisor@rtbrick>SPINE01: cfg>
```

LDP Timer Configuration

Specify the hello timer and hold-down timer for LDP adjacency. Similarly, specify the keepalive and keepalive timeout settings for the LDP session.

Syntax:

set instance <instance-name> **protocol ldp timer** <attribute> <value>

Attribute	Description
hello hold-time <hold-time>	Specifies the hello hold-time interval in seconds before declaring a neighbor to be down. Range: 0-65535. Default: 15.
hello interval <interval>	Specifies the hello messages interval in seconds. Range: 0-65535. Default: 5.
session keepalive-interval <keepalive-interval>	Specifies the session keepalive messages interval in seconds. Range: 1-65535. Default: 10.
session keepalive-timeout <keepalive-timeout>	Specifies the session keepalive timeout in seconds before declaring a session to be down. Range: 1-65535. Default: 30.

Example 1: LDP Timer Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp timer
{
    "rtbrick-config:timer": {
      "hello": {
        "interval": 10,
        "hold-time": 20
      },
      "session": {
        "keepalive-interval": 3000,
        "keepalive-timeout": 5000
      }
    }
  }
supervisor@rtbrick>SPINE01: cfg>
```

LDP IGP Synchronization

Synchronization between LDP and the underlying interior gateway protocol (IGP) ensures that the LDP path is fully established before the IGP path is used for forwarding traffic. LDP IGP synchronization is supported only on interfaces running Intermediate System-to-System (IS-IS) or OSPFv2 processes.

Syntax:

set instance <instance-name> **protocol ldp igp-synchronization** <attribute> <value>

Attribute	Description
hold-timer <hold-timer>	Specifies the hold-timer in seconds to limit how long the IGP session must wait before declaring the LDP synchronization. Range: 0-60. Default: 10.

Example 1: LDP IGP Synchronization Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp igp-
synchronization
{
    "rtbrick-config:igp-synchronization": {
     hold-timer": 60,
    }
  }
supervisor@rtbrick>SPINE01: cfg>
```

LDP Authentication Configuration

To meet the security requirements of LDP sessions, configure LDP authentication.

Syntax:

set instance <instance-name> **protocol ldp session ipv4|ipv6** <address> **authentication-id** <authentication-id>

Attribute	Description
<address>	Specifies the transport IP address of the peer.
<authentication-id>	Authentication Tuple Identifier

Example 1: LDP Authentication Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp session
ipv4
{
  "rtbrick-config:ipv4": [
    {
      "address": "198.51.100.2",
      "authentication-id": "auth_id_1"
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

TCP Authentication Configuration

In the instance TCP authentication hierarchy, you can optionally enable MD5 or HMAC-SHA-1-96, HMAC-SHA-256-128, or AES-128-CMAC-96 authentication. Authentication is not configured for LDP directly, but for the TCP sessions used by LDP.

Syntax:

set instance <instance> **tcp authentication** <authentication-id> <attribute> <value>

Attribute	Description
<authentication-id>	Authentication identifier
type <type>	Authentication identifier such as MD5 or HMAC-SHA-1-96,HMAC-SHA-256-128 or AES-128-CMAC-96
key1-id <key1-id>	Key ID1 of the receiver
key1-encrypted-text <key1-encrypted-text>	Encrypted text of key1
key1-plain-text <key1-plain-text>	Plain text of key1
key2-id <key2-id>	Key ID2 of the receiver
key2-encrypted-text <key2-encrypted-text>	Encrypted text of key2
key2-plain-text <key2-plain-text>	Plain text of key2

Example: LDP TCP Authentication Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default tcp authentication
auth_id_1
{
  "rtbrick-config:authentication": [
    {
      "authentication-id": "auth_id_1",
      "type": "MD5",
      "key1-id": 1,
      "key1-encrypted-text": "$2a6fd7db50a18a9f1f16b5c5b4214fab0"
    }
  ]
}
```

```
supervisor@rtbrick>SPINE01: cfg>
```

LDP Import and Export Policy Configuration

Syntax:

set instance <instance-name> **protocol ldp session ipv4|ipv6** <address> <attribute> <value>

Attribute	Description
<address>	Specifies the IPv4 or IPv6 address.
export-policy <export-policy>	Export policy identifier
import-policy <import-policy>	Import policy identifier

Example 1: LDP Export Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp session
ipv4
{
  "rtbrick-config:ipv4": [
    {
      "address": "198.51.100.2",
      "export-policy": "exp-policy1"
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

Example 3: LDP Import Configuration

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol ldp session
ipv4
{
  "rtbrick-config:ipv4": [
    {
      "address": "198.51.100.2",
      "import-policy": "imp-policy1"
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

2.8.3. LDP Operational Commands

LDP Show Commands

The LDP show commands provide detailed information about the LDP protocol operations.

LDP Summary

Syntax:

show ldp summary <options>

Option	Description
-	Without any option, the command displays the LDP summary information for all instances.
instance <instance-name>	Displays LDP summary information about the specified instance.

Example: LDP summary for the default instance

```
supervisor@rtbrick>SPINE01: op> show ldp summary
Instance: default
  General information:
    LDP identifier: 198.51.100.2:0, Version: 1
    FEC resolution: Best match
    Protocol preference: 9
    LSR ID: 198.51.100.2
    IPv4 Status: True
    IPv6 Status: True
  Modes:
    Advertisement mode: Downstream Unsolicited
    Advertisement control mode: Ordered
    Label retention mode: Liberal
  Capabilities:
    IPv6 address family: -  , Graceful restart: False
    Loop detection: False
      Hop count: -, Vector length: -
  Timers:
    Adjacency:
      Hello: 5s, Holdtime: 15s
    Targeted adjacency:
      Hello: 15s, Holdtime: 45s
    Session:
      Keepalive: 10s, Holdtime: 30s
  Statistics:
    Adjacency:
      Link adjacency: 2, Targeted adjacency: 0
    Session:
      Session in non-existent: 0, Session in initialized: 0
      Session in opensent: 0, Session in openconfirm: 0
      Session in operational: 2
Instance: green
  General information:
    LDP identifier: 192.0.2.2:0, Version: 1
    FEC resolution: Best match
    Protocol preference: 9
    LSR ID: 192.0.2.2
    IPv4 Status: True
    IPv6 Status: True
  Modes:
    Advertisement mode: Downstream Unsolicited
    Advertisement control mode: Ordered
    Label retention mode: Liberal
  Capabilities:
    IPv6 address family: -  , Graceful restart: False
    Loop detection: False
      Hop count: -, Vector length: -
  Timers:
    Adjacency:
      Hello: 5s, Holdtime: 15s
    Targeted adjacency:
      Hello: 15s, Holdtime: 45s
    Session:
      Keepalive: 10s, Holdtime: 30s
<...>
```

To access the Operational State API that corresponds to this CLI, click here.

LDP Neighbor

Syntax:

show ldp neighbor <options>

Option	Description
-	Without any option, this command displays information about LDP neighbors.
detail	Detailed information about the LDP neighbors.
instance <instance-name>	Displays LDP neighbor information about the specified instance.
instance <instance-name> detail	Displays detailed LDP neighbor information about the specified instance.

Option	Description
instance <instance-name> ldp-id <ldp-id>	Displays LDP neighbor information about the specified LDP identifier and instance.
interface <name>	Displays LDP neighbor information about the specified interface.
interface <name> detail	Displays detailed LDP neighbor information about the specified interface.
ldp-id <ldp-id>	Displays LDP neighbor information about the specified LDP identifier.

Example 1: Summary view of all the LDP neighbors

```
supervisor@rtbrick>SPINE01: op> show ldp neighbor
Instance: default
  Interface        LDP ID              Transport IP   Up Since
Expires
  ifl-0/0/0/1      198.51.100.2:0      198.51.100.2   Thu Feb 09 12:17:15       in
11s
  ifl-0/0/2/1      198.51.100.3:0      198.51.100.3   Thu Feb 09 12:17:31       in
12s
  ifl-0/0/0/100    198.51.100.2:0      198.51.100.2   Thu Feb 09 12:17:15       in
11s
  ifl-0/0/0/101    198.51.100.2:0      198.51.100.2   Thu Feb 09 12:17:15       in
11s
  ifl-0/0/1/102    198.51.100.2:0      198.51.100.2   Thu Feb 09 12:17:15       in
11s
supervisor@rtbrick>SPINE01: op>
```

Example 2: Detailed view of all the LDP neighbors

```
supervisor@rtbrick>SPINE01: op> show ldp neighbor detail
Instance: default
  LDP-Identifier: 198.51.100.1:0, Peer: 198.51.100.1, Type: Targeted
  Negotiated holdtime: 45000, Expiry time: 31s 880316us
  Local link address: 198.51.100.2, Peer link address: 198.51.100.1
  Local transport address: 198.51.100.2, Peer transport address: 198.51.100.1
  Local holdtime: 45, Peer holdtime: 45, Up since: Tue Mar 04 05:59:47
  Local transport preference : ipv6, Peer transport preference : ipv6
  Last transition time: Tue Mar 04 07:05:47 GMT +0000 2025
Instance: default
  LDP-Identifier: 198.51.100.5:0, Peer: 198.51.100.5, Type: Targeted
  Negotiated holdtime: 45000, Expiry time: 40s 850139us
  Local link address: 198.51.100.2, Peer link address: 198.51.100.5
  Local transport address: 198.51.100.2, Peer transport address: 198.51.100.5
  Local holdtime: 45, Peer holdtime: 45, Up since: Tue Mar 04 06:00:11
  Local transport preference : ipv6, Peer transport preference : ipv6
  Last transition time: Tue Mar 04 07:05:56 GMT +0000 2025
Instance: default
  LDP-Identifier: 198.51.100.1:0, Peer: 2001:DB8::1, Type: Targeted
  Negotiated holdtime: 45000, Expiry time: 31s 880318us
  Local link address: 2001:DB8::2, Peer link address: 2001:DB8::1
  Local transport address: 2001:DB8::2, Peer transport address: 2001:DB8::1
  Local holdtime: 45, Peer holdtime: 45, Up since: Tue Mar 04 05:59:47
  Local transport preference : ipv6, Peer transport preference : ipv6
  Last transition time: Tue Mar 04 07:05:47 GMT +0000 2025
<...>
```

Example 3: Summary view of all the targeted LDP neighbors

```
supervisor@rtbrick>SPINE01: op> show ldp neighbor
Instance: default
  Interface/Peer     LDP ID              Transport IP      Up Since
Expires
  198.51.100.1       198.51.100.1:0      198.51.100.1      Tue Mar 04 05:59:47
in 38s
  198.51.100.5       198.51.100.5:0      198.51.100.5      Tue Mar 04 06:00:11
in 32s
  2001:DB8::1        198.51.100.1:0      2001:DB8::1       Tue Mar 04 05:59:47
in 38s
  2001:DB8::5        198.51.100.5:0      2001:DB8::5       Tue Mar 04 05:59:56
in 32s
Instance: green
  Interface/Peer     LDP ID              Transport IP      Up Since
Expires
  192.0.2.1          192.0.2.1:0         192.0.2.1         Tue Mar 04 05:59:47
in 38s
  192.0.2.5          192.0.2.5:0         192.0.2.5         Tue Mar 04 06:00:11
in 32s
supervisor@rtbrick>SPINE01: op>
```

Example 4: Detailed view of a targeted LDP neighbor

```
supervisor@rtbrick>SPINE01: op> show ldp neighbor ldp-id 198.51.100.1:0
Instance: default
  LDP-Identifier: 198.51.100.1:0, Peer: 198.51.100.10, Type: Targeted
  Negotiated holdtime: 45000, Expiry time: 35s 775898us
  Local link address: 198.51.100.2, Peer link address: 198.51.100.1
  Local transport address: 198.51.100.2, Peer transport address: 198.51.100.1
  Local holdtime: 45, Peer holdtime: 45, Up since: Tue Mar 04 05:59:47
  Local transport preference : ipv6, Peer transport preference : ipv6
  Last transition time: Tue Mar 04 07:04:02 GMT +0000 2025
Instance: default
  LDP-Identifier: 198.51.100.1:0, Peer: 2001:DB8::1, Type: Targeted
  Negotiated holdtime: 45000, Expiry time: 35s 775923us
  Local link address: 2001:DB8::2, Peer link address: 2001:DB8::1
  Local transport address: 2001:DB8::2, Peer transport address: 2001:DB8::1
  Local holdtime: 45, Peer holdtime: 45, Up since: Tue Mar 04 05:59:47
  Local transport preference : ipv6, Peer transport preference : ipv6
  Last transition time: Tue Mar 04 07:04:02 GMT +0000 2025
supervisor@rtbrick>SPINE01: op>
```

LDP Interface

Syntax:

show ldp interface <option>

Option	Description
-	Without any option, this command displays information on all configured LDP interfaces.
<interface-name>	Specify the name of the interface.
detail	Detailed interface information.

Example 1: Summary view of the specified LDP interface

```
supervisor@rtbrick: cfg> show ldp interface ifl-0/0/1/1
Instance: green
  Interface: ifl-0/0/1/1
    Primary IPv4 Address: 192.0.2.102, Primary IPv6 Address: 2001:DB8:12::1
    Session Hold: 30, Session Keepalive: 10
    Discovery Hello: 5, Discovery Hold: 15
    Neighbor count: 1, Transport preference: ipv4
    IPv4 enable: True, IPv6 enable: True, Cisco interop: False
```

Example 2: Summary view of all the LDP interfaces

```
supervisor@rtbrick: cfg> show ldp interface
Instance: default
  Interface             Primary IPv4 Address      Primary IPv6 Address
  ifl-0/0/0/1           198.51.100.100            2001:DB8:1::1
  ifl-0/0/0/100         198.51.100.101            2001:DB8:2::1
  ifl-0/0/0/101         198.51.100.102            2001:DB8:3::1
  ifl-0/0/1/102         198.51.100.103            2001:DB8:4::1
  ifl-0/0/2/1           198.51.100.104            2001:DB8:5::1
  ifl-0/0/3/1           198.51.100.105            2001:DB8:6::1
  lo-0/0/0/1            198.51.100.1
  lo-0/0/0/2            198.51.100.2
  lo-0/0/0/3            198.51.100.3
  lo-0/0/0/4            198.51.100.4
  lo-0/0/0/5            198.51.100.5
Instance: green
  Interface             Primary IPv4 Address      Primary IPv6 Address
  ifl-0/0/0/202         192.0.2.100               2001:DB8:11::1
  ifl-0/0/1/1           192.0.2.102               2001:DB8:12::1
  ifl-0/0/1/200         192.0.2.103               2001:DB8:13::1
  ifl-0/0/1/201         192.0.2.104               2001:DB8:14::1
  ifl-0/0/2/200         192.0.2.105               2001:DB8:15::1
  ifl-0/0/3/200         192.0.2.105               2001:DB8:16::1
  lo-0/0/1/1            192.0.2.1
  lo-0/0/1/2            192.0.2.2
  lo-0/0/1/3            192.0.2.3
  lo-0/0/1/4           192.0.2.4
  lo-0/0/1/5           192.0.2.5
```

LDP Session

Syntax:

show ldp session <options>

Option	Description
-	Without any option, this command displays a summary of LDP session information.
detail	Displays detailed information about the LDP sessions.
instance <instance-name>	Displays LDP session information about the specified instance.
instance <instance-name> detail	Displays detailed LDP session information about the specified instance.
instance <instance-name> ldp-id <ldp-id>	Displays LDP session information about the specified LDP identifier and instance.
ldp-id <ldp-id>	Displays LDP session information about the specified LDP identifier.

Example 1: Summary view of the LDP session

```
supervisor@rtbrick>SPINE01: op> show ldp session
Instance: default
  LDP ID            Peer IP           State             Up/Down           FECRcvd   FECSent
  198.51.100.2:0    198.51.100.2       Operational       0d:00h:29m:44s        15        15
  198.51.100.3:0    198.51.100.3       Operational       0d:00h:29m:29s        15        15
supervisor@rtbrick>SPINE01: op>
```

Example 2: Detailed view of the LDP session

```
supervisor@rtbrick>SPINE01: op> show ldp session detail
Instance: default
  LDP Identifier: 198.51.100.2:0, Peer IP: 198.51.100.2, Local IP: 198.51.100.1
    Type: link, State: Operational, Uptime: 0d:00h:34m:35s
    Reason:
    Last transition: Thu Feb 09 12:17:28 GMT +0000 2023, Flap count: 0
  Advertisement Mode:
    Peer: Downstream unsolicited, Local: Downstream unsolicited
    Negotiated: Downstream unsolicited
  Timers:
    Connect retry: 10s
```

```
      Peer keepalive interval: 10s, Local keepalive interval: 10s
      Peer keepalive timeout: 30s, Local keepalive timeout: 30s
      Negotiated keepalive interval: 10s
      Negotiated keepalive timeout: 30s
    Received messages:
      Initialization: 1, KeepAlive: 208, Notification: 0
      Address: 1, Address Withdraw: 0, Label Mapping: 15
      Label Withdraw: 0, Label Release: 0
    Sent messages:
      Initialization: 1, KeepAlive: 208, Notification: 0
      Address: 1, Address Withdraw: 0, Label Mapping: 15
      Label Withdraw: 0, Label Release: 0
    Capability:
      Typed WildCard FEC:
      Local Support: True, Peer Support: True, Negotiated: True
    Total received messages:
      Initialization: 1, KeepAlive: 92, Notification: 0
      Address: 2, Address Withdraw: 0, Label Mapping: 20
    Total sent messages:
      Initialization: 1, KeepAlive: 92, Notification: 0
      Address: 2, Address Withdraw: 0, Label Mapping: 20
      Label Withdraw: 0, Label Release: 0
  <...>
```

In the case of a T-LDP session, the "Type" will be "Targeted".

To access the Operational State API that corresponds to this CLI, click here.

LDP Address

Syntax:

show ldp address <options>

Option	Description
-	Without any option, this command displays a summary of all the interface addresses received from the LDP sessions.
instance <instance-name>	Displays LDP address information about the specified instance.
instance <instance-name> <afi>	Displays LDP address of the specified address family (AFI). Supported values: ipv4, ipv6.
instance <instance-name> ldp-id <ldp-id>	Displays LDP address information about the specified LDP identifier and instance.

Option	Description
ldp-id <ldp-id>	Displays LDP address information about the specified LDP identifier.

Example: Summary view of all the LDP addresses

```
supervisor@rtbrick>SPINE01: op> show ldp address
Instance: default, LDP Identifier: 198.51.100.2:0, AFI: ipv4
  198.51.100.61
  198.51.100.102
  198.51.100.63
  198.51.100.94
  198.51.100.2
  198.51.100.65
  198.51.100.222
  198.51.100.21
  198.51.100.214
  198.51.100.48
<...>
```

LDP Binding

Syntax:

show ldp binding <options>

Option	Description
-	Without any option, this command displays a summary of all the LDP label bindings.
instance <instance-name>	Displays LDP label binding information about the specified instance.
instance <instance-name> prefix <ip>	Displays LDP label binding information about the specified prefix and instance. Supported prefix values: ipv4, ipv6.
prefix <ip>	Displays the LDP label binding information for the specified prefix. Supported prefix values: ipv4, ipv6.
received	Displays the LDP received label binding information of the LDP sessions.
received instance <instance-name>	Displays LDP received label binding information of the specified instance.

Option	Description
received instance <instance-name> ldp-id <ldp-id>	Displays LDP received label binding information about the specified LDP identifier and instance.
received ldp-id <ldp-id>	Displays LDP received label binding information of the specified LDP identifier.
sent	Displays the LDP sent label binding information of the LDP sessions.
sent instance <instance-name>	Displays LDP sent label binding information of the specified instance.
sent instance <instance-name> ldp-id <ldp-id>	Displays LDP sent label binding information about the specified LDP identifier and instance.
sent ldp-id <ldp-id>	Displays LDP sent label binding information of the specified LDP identifier.

Example 1: Summary view of the LDP binding

```
supervisor@rtbrick>SPINE01: op> show ldp binding

Instance: default, AFI: ipv4
  Prefix                  In Label           Out Label          LDP ID
Status
  198.51.100.1/24          -                  label:3            -
Best
                          label:20066        -                  198.51.100.3:0
Non-best
                          label:20065        -                  198.51.100.2:0
Non-best
  198.51.100.11/24         -                  label:3            -
Best
                          label:20066        -                  198.51.100.3:0
Non-best
                          label:20065        -                  198.51.100.2:0
Non-best
  198.51.100.41/24         -                  label:3            -
Best
                          label:20066        -                  198.51.100.3:0
Non-best
                          label:20065        -                  198.51.100.2:0
Non-best
  198.51.100.44/24         -                  label:3            -
Best
                          label:20066        -                  198.51.100.3:0
Non-best
                          label:20065        -                  198.51.100.2:0
Non-best
  198.51.100.47/24         -                  label:3            -
Best
```

```
                        label:20066        -                  198.51.100.3:0
Non-best
                        label:20065        -                  198.51.100.2:0
Non-best
  198.51.100.2/24        label:3            label:20065       198.51.100.2:0
Best
                        label:20065        -                  198.51.100.3:0
Non-best
  198.51.100.21/24        label:3            label:20065      198.51.100.2:0
Best
                        label:20065        -                  198.51.100.3:0
Non-best
  198.51.100.42/24        label:3            label:20065      198.51.100.2:0
Best
                        label:20065        -                  198.51.100.3:0
Non-best
<...>
```

Example 2: Summary view of the LDP binding for the specified prefix

```
supervisor@rtbrick>SPINE01: op> show ldp binding prefix 198.51.100.2/24
Instance: default, AFI: ipv4
  Prefix                In Label           Out Label          LDP ID
Status
  198.51.100.2/24       label:3            label:20065        198.51.100.2:0
Best
                        label:20065        -                  198.51.100.3:0
Non-best
supervisor@rtbrick>SPINE01: op>
```

LDP Route

Syntax:

show ldp route <options>

Option	Description
-	Without any option, this command displays a summary of LDP route information.
instance <instance-name>	Displays LDP route information for the specified instance.
instance <instance-name> <afi>	Displays LDP route information for the specified address family and instance. Supported AFI values: ipv4, ipv6, and mpls.
instance <instance-name> ipv4 prefix <ip>	Displays LDP route information for the specified address family of IPv4 prefix and instance.

Option	Description
instance <instance-name> ipv6 prefix <ip>	Displays LDP route information for the specified address family of IPv6 prefix and instance.
instance <instance-name> prefix <ip>	Displays LDP route information for the specified prefix and instance.
instance <instance-name> label <label>	Displays LDP route information for the specified MPLS label and instance.
instance <instance-name> mpls	Displays LDP route information about MPLS labels.
instance <instance-name> mpls label <label>	Displays LDP route information for the specified MPLS label and instance.
label <label>	Displays LDP route information for the specified MPLS label.
ipv4	Displays LDP route information about the IPv4 address family.
ipv4 prefix <ip>	Displays LDP route IPv4 address family information for the specified prefix.
ipv6	Displays LDP route information about the IPv6 address family.
ipv6 prefix <ip>	Displays LDP route IPv6 address family information for the specified prefix.
mpls	Displays LDP route information about MPLS labels.
mpls label <label>	Displays LDP route information for the specified MPLS label.
prefix <ip>	Displays LDP route information for the specified prefix address.

Example: Summary view of the LDP route

```
supervisor@rtbrick>SPINE01: op> show ldp route
Instance: default, AFI: ipv4, SAFI: labeled-unicast
  Prefix/Label         Advertised label  Received label    Next Hop
Interface           Metric
  198.51.100.1/24       3                 -                 -               -
-
  198.51.100.2/24       20065             -                 198.51.100.61
ifl-0/0/0/1      1000000
  198.51.100.3/24       20067             20067             198.51.100.61
```

```
ifl-0/0/0/1      2000001
  198.51.100.11/24       3                  -                  -              -
-
  198.51.100.21/24       20065              -                  198.51.100.61
ifl-0/0/0/1      1000000
  198.51.100.31/24       20067              20067              198.51.100.61
ifl-0/0/0/1      2000001
  198.51.100.41/24       3                  -                  -              -
-
  198.51.100.42/24       20065              -                  198.51.100.61
ifl-0/0/0/1      1000000
  198.51.100.43/24       20067              20067              198.51.100.61
ifl-0/0/0/1      2000001
  198.51.100.44/24       3                  -                  -              -
-
  198.51.100.45/24       20065              -                  198.51.100.61
ifl-0/0/0/1      1000000
  198.51.100.46/24       20067              20067              198.51.100.61
ifl-0/0/0/1      2000001
  198.51.100.47/24       3                  -                  -              -
-
  198.51.100.48/24       20065              -                  198.51.100.61
ifl-0/0/0/1      1000000
  198.51.100.49/24       20067              20067              198.51.100.61
ifl-0/0/0/1      2000001
<...>
```

LDP Statistics

Syntax:

show ldp statistics <options>

Option	Description
-	Without any option, the command displays the LDP statistics for all instances.
instance <instance-name>	Displays LDP statistics information about the specified instance.
instance <instance-name> ldp-id <ldp-id>	Displays LDP statistics information about the specified LDP identifier and instance.

Example: Summary view of the LDP statistics

```
supervisor@rtbrick>SPINE01: op> show ldp statistics
Instance: default, LDP ID: 198.51.100.2:0
  Received messages:
    Initialization: 1, KeepAlive: 558, Notification: 0
    Address: 1, Address Withdraw: 0, Label Mapping: 15
    Label Withdraw: 0, Label Release: 0
  Sent messages:
    Initialization: 1, KeepAlive: 558, Notification: 0
    Address: 1, Address Withdraw: 0, Label Mapping: 15
    Label Withdraw: 0, Label Release: 0
Instance: default, LDP ID: 198.51.100.3:0
  Received messages:
    Initialization: 1, KeepAlive: 557, Notification: 0
    Address: 1, Address Withdraw: 0, Label Mapping: 15
    Label Withdraw: 0, Label Release: 0
  Sent messages:
    Initialization: 1, KeepAlive: 557, Notification: 0
    Address: 1, Address Withdraw: 0, Label Mapping: 15
    Label Withdraw: 0, Label Release: 0
supervisor@rtbrick>SPINE01: op>
```

LDP TCP connection

Syntax:

show ldp tcp connection <options>

Option	Description
-	Without any option, the command displays the TCP connections used by LDP for all instances.
detail	Detailed list view of the TCP connections.
detail instance <instance-name>	Detailed list view of the TCP connections of the specified instance.
instance <instance-name>	TCP connections summary of the specified instance.

Example: Summary view of the LDP TCP connections

```
supervisor@rtbrick>SPINE01: op> show ldp tcp connection
Instance       Local IP             Remote IP            Local port     Remote
port   State
default        198.51.100.1         198.51.100.2                646
64718   Established
default        198.51.100.1         198.51.100.3                646
64718   Established
supervisor@rtbrick>SPINE01: op>
```

LDP Clear Commands

Clear commands allow resetting operational states.

Clear LDP Session

Syntax:

clear ldp session <options>

Option	Description
all	Clears all the LDP sessions.
all soft-in	Sends route refresh to all neighbors to receive FEC bindings.
all soft-out	Re-advertises all the routes previously sent to the peers.
instance <instance-name> all	Clears all the LDP sessions for the specified instance.
instance <instance> all soft-in	Sends route refresh to all neighbors to receive FEC bindings for the specified instance.
instance <instance> all soft-out	Re-advertises all the routes previously sent to the peers for the specified instance.
instance <instance-name> peer ldp-id <ldp-id>	Clears the LDP session for the specified instance and peer LDP identifier.
instance <instance> peer ldp-id <ldp-id> soft-in	Sends route refresh to the specific peer to receive FEC bindings for the specified instance and peer ldp-id.
instance <instance> peer ldp-id <ldp-id> soft-out	Re-advertises all the routes previously sent to the specific peer for the specified instance and peer ldp-id.

Example: The example below shows how to clear all the LDP sessions.

```
supervisor@rtbrick>SPINE01: op> clear ldp session all
LDP session cleared with instance default
supervisor@rtbrick>SPINE01: op>
```

To access the Operational State API that corresponds to this CLI, click here.

Clear LDP Statistics

Syntax:

clear ldp statistics <options>

Option	Description
all	Clears all the LDP statistics.
instance <instance-name> all	Clears all the LDP statistics for the specified instance.
instance <instance-name> peer ldp-id <ldp-id>	Clears the LDP statistics for the specified instance and peer LDP identifier.

Example: The example below shows how to clear all the LDP statistics.

```
supervisor@rtbrick>SPINE01: op> clear ldp statistics all
LDP statistics cleared for instance default
supervisor@rtbrick>SPINE01: op>
```

Clear LDP Neighbor

Syntax:

clear ldp neighbor <options>

Option	Description
all	Clears all the LDP neighbors.
instance <instance-name>	Clears the LDP neighbor for the specified instance.

Example: The example below shows how to clear all the LDP neighbor.

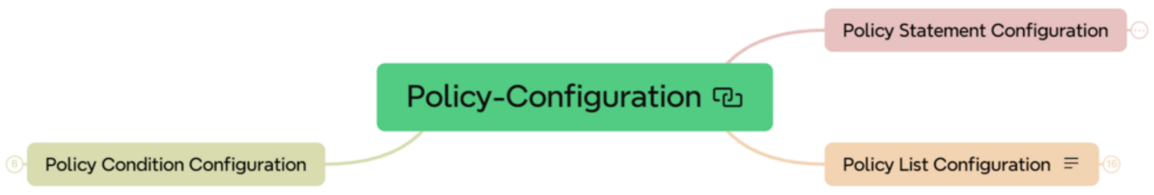
```
supervisor@rtbrick>SPINE01: op> clear ldp neighbor all
LDP neighbor cleared with instance default
supervisor@rtbrick>SPINE01: op>
```

2.9. Policy

2.9.1. Policy Overview

Policies are rules that allow to control and modify the behavior of routing protocols such as BGP, IS-IS, OSPF, and other supported protocols. The policy framework is generic; it serves multiple purposes and applications. Policies are first created using a common policy configuration and then applied by attaching them to an application like a protocol.

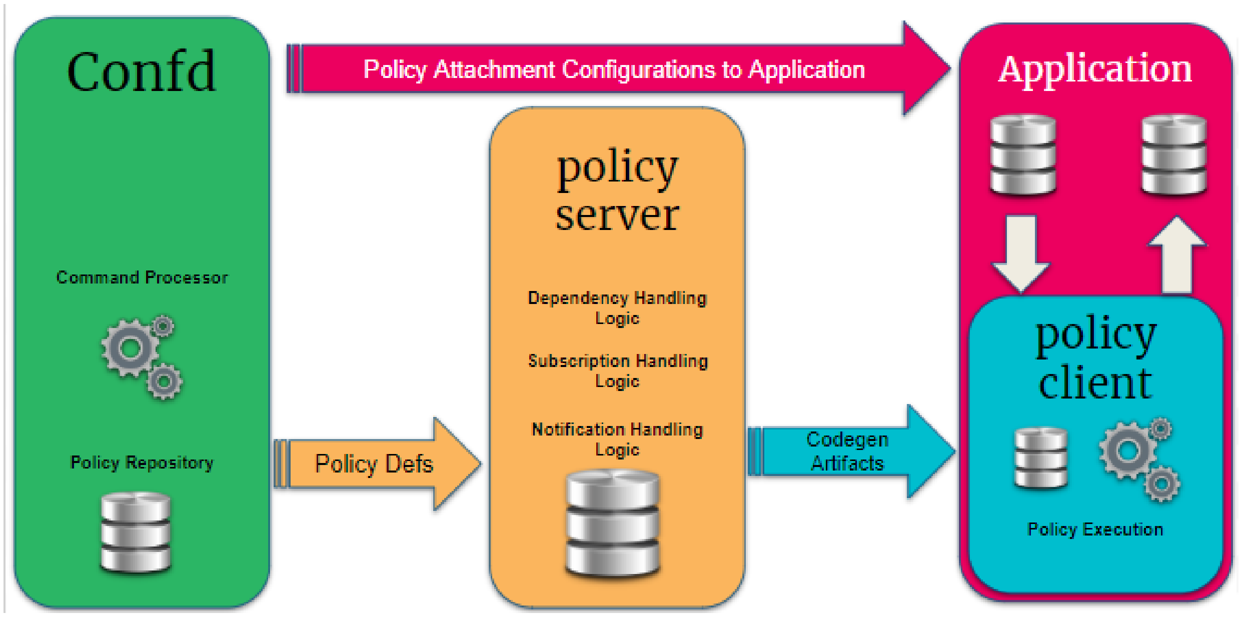
The following diagram shows policy configuration options.


Policy-Configuration
Policy Statement Configuration
Policy Condition Configuration
Policy List Configuration

Policy Components

In RtBrick Full Stack, the policy implementation consists of 4 sub-components:

- Policy Repository
- Command Processing Module
- Policy Server, the policy generation and relationship management component
- Policy Client, the policy enforcement component


Confd
Command Processor
Policy Repository
Policy Attachment Configurations to Application
Policy Defs
policy server
Dependency Handling Logic
Subscription Handling Logic
Notification Handling Logic
Codegen Artifacts
Application
policy client
Policy Execution

Policy Repository

The policy repository contains all tables related to policy and the associated list of match criteria.

Command Processing Module

The command processing module is part of the configuration daemon (confd), and it handles user interaction with the policy module. This is the back-end of the Command Line Interface (CLI) and JSON configuration that supports the policy configurations.

This module maps the user-defined configuration into the back-end policy object, which is used by the execution engine (after verification), and it ensures that the policy can be correctly executed. This module relays the user intent via related BDS tables to the policy server.

Policy Server

The policy server is a server component that manages all policy rules in the various policy tables and also code generation of the policies.

The following are the functionalities of the policy server:

- Parses the objects in the policy tables and is an execution engine that generates the code to build the policy rules for evaluation, the relationship between various objects, and relays the intent to the evaluation engine.
- Maintains relationships between various policy constructs such as policy statements, rules, ordinals, and lists.
- Tracks the attachment points so that when policies are modified, the appropriate clients are notified of the relevant new policies.
- Flattens the various relationships and generates a notification table that the clients subscribe to obtain notifications based on specific interest groups.
- Uses dependency table relationships to generate jobs to trigger code generation for various policy components.
- On code generation, the policy server updates a notification table that maintains the mapping between the policy server and the client interest groups. The notification table is a single point for the dissemination of information so that it can generate notifications for clients depending on their

subscriptions for policy of interest.

- Policy server notification is generated for the policy clients. A notification is received from the notification table with metadata information that notifies the client if this is a new version of the policy or the original version of the policy. The client uses this information to enforce the policy evaluation and to decide on the version of the policy rule that should be used.

Policy Client

The policy client is a shared library component to which a client daemon, like BGP, IS-IS, OSPF, etc., links. This is the component that performs policy enforcement. It performs the following tasks:

- Links with client daemons like BGP, IS-IS, and OSPF.
- Contains a listener that gets notifications on the availability of a new policy rule that is generated by the policy server.
- Evaluates the compiled rule, and if there are any listeners/ interests, then notifies the components within the client daemon.
- Evaluates any policy configurations on the client daemon and invokes policy processing in response.

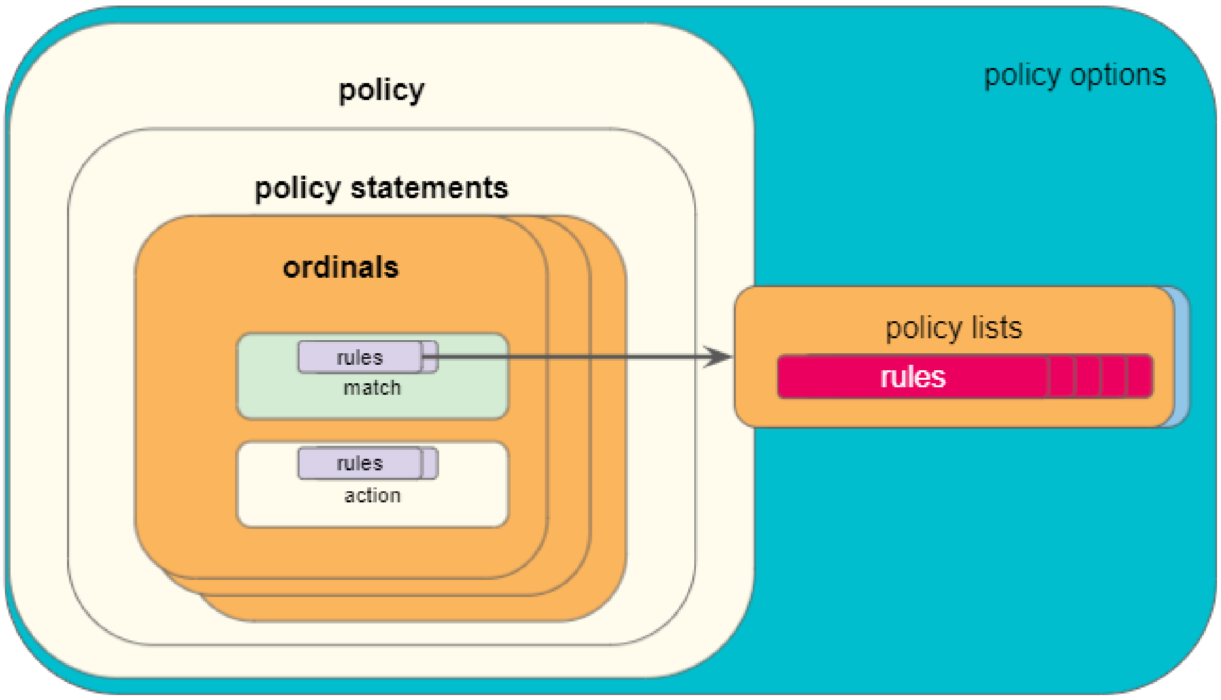
Tables and Subscriptions

The table below shows the various tables and their sharing across various policy components.

Confd	global.policy.list.config global.policy.list.entry.config global.policy.match.rules.config global.policy.statement.config global.policy.ordinal.config global.policy.mapping.list global.policy.mapping.rules	Policy Statement is composed of one or more policy terms. Each term has a match action criteria. In the match and action criteria, either a single element or a list of elements are compared, and actions are taken. The actions include accept, deny, flow-control, etc.
policy.server	global.policy.dependency global.<bds_name>.policy.subscription global.<bds_name>.policy.notification	Policy Server subscribes to all the tables from confd and creates tables that track policy-entry and dependency and notifies clients after code generation.
policy.client	global.<bds_name>.policy.shared.object.cache global.<bds_name>.policy.subscription global.<bds_name>.policy.context	Subscribes to code generation notifications application context and maintains cache of subscribed .so

Policy Building Blocks

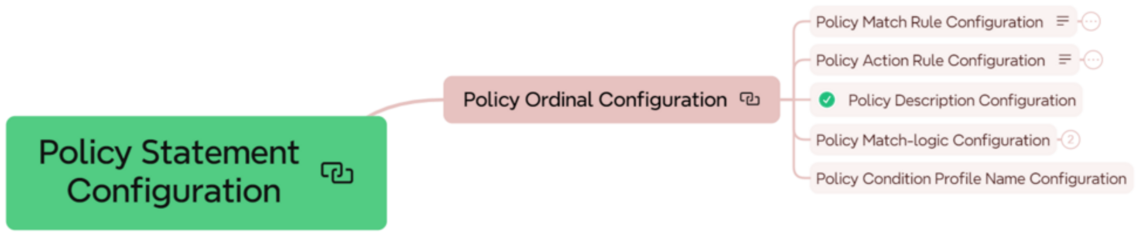
The figure below shows the building blocks of a policy.



Statements

A policy is defined by a policy statement. A policy statement is a compound block of policy definition that consists of one or more set of rules called ordinals. A policy statement is exercised in the order defined. The statement name is a globally unique string that is used to identify the policy, and used by the applications.

The following diagram shows policy statement configuration parameters.


Policy Statement Configuration
Policy Ordinal Configuration
Policy Match Rule Configuration
Policy Action Rule Configuration
Policy Description Configuration
Policy Match-logic Configuration
Policy Condition Profile Name Configuration

Ordinals

A policy ordinal is the smallest block to represent a user policy intent and consists of rules for match and action blocks. The match blocks can either define single independent elements like AS path, IP prefix, IP addresses, community, and so on, or a list of the elements maintained in a different table.

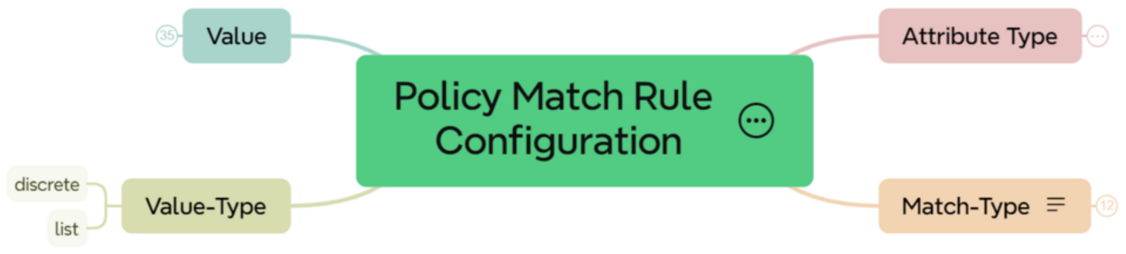
- An ordinal must be a unique number within the scope of a statement which determines the order of the term execution within a policy statement.

- If no ordinal exist or configured, and if the policy is used, then all routes or objects will be denied.
- A match logic is defined per ordinal. In case of multiple match rules, it defines if all rules (and), or any of the rules (or) have to match.

Match Rules

Match rules define criteria to evaluate and select routes or other objects to which the policy is applied. One or more match rules compose a match block.

The following diagram shows policy match rule parameters.



- If the match block results in a successful match ("true"), the corresponding action block will executed.
- If the match block result is unsuccessful ("false"), the action block will not be executed, and the next ordinal will be processed.
- If there is no match block, the action block will be executed.

In case of multiple match rules, the behaviour depends on the configured match logic:

- If the match logic is or, the match block result will be succcessful ("true") if any one rule matches. Otherwise, by default it is "false".
- If the match logic is and, the match block result will be successful ("true") if all rules match. Otherwise it is "false".

A match rule can refer to a single discrete value, or a list. A policy list is configured separately and referenced from the policy statement. In case of a list match, the behaviour is as follows:

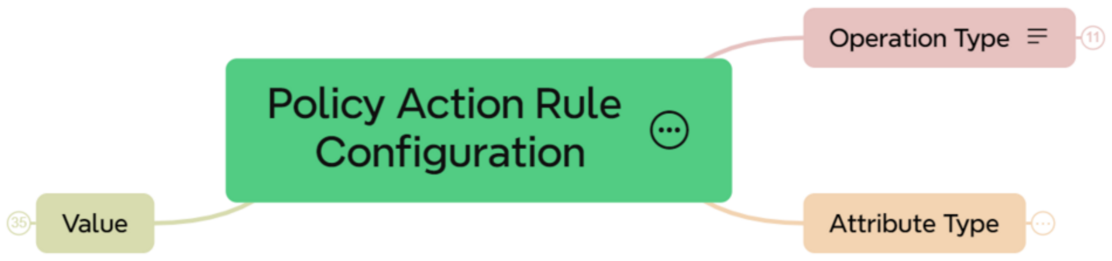
- If any of the list entries matches the configured value, the match block result will be succcessful ("true"). Otherwise it is "false".
- If the list is defined but empty, the match rule result will be unsuccessful ("false").

Action Rules

Action rules define operations like return-permit or return-deny, or flow control commands like goto-next-ordinal. The action block will be executed if there is a successful match. Otherwise, the next ordinal is processed. The action block can contain one or multiple action rules. In case of multiple action rules, all actions will be applied. The rules are executed in the order of the rule numbers. If there is any termination action, the rules afterwards are not executed anymore.

The action block is optional. The implicit default action is return-permit.

The following diagram shows action rules configuration parameters.



Lists

A policy list is a list of values that can be referenced by a match rule in a policy statement. If you have a number of values like for example route prefixes, it is more efficient to refer to a policy list instead of creating one match rule per prefix.

Policy lists are configured separately and can thereby be maintained more easily. Besides, policy lists can be referenced by multiple policy statements.

Conditions

Policy conditions are configured separately and can be used as an additional option in policy statements. A policy rule (ordinal) will only be executed if the condition is true. Conditional policies allow to make policy execution depended on certain states of the system, for example:

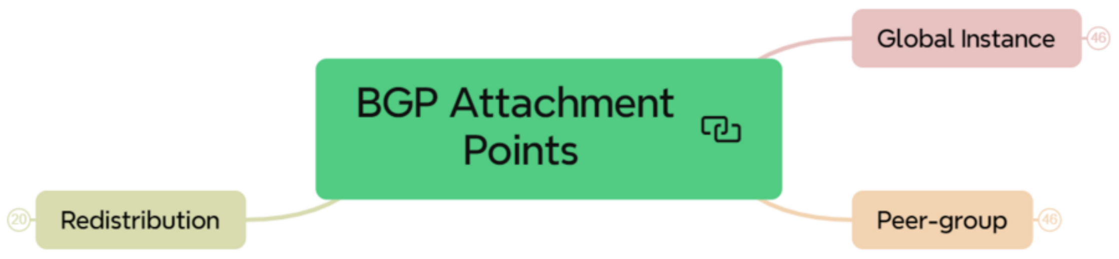
- Protocol neighbor states
- Presence or absence of a specific route and/or path attribute
- Number of routes in a routing table

One condition is supported per ordinal. A single condition can be attached to multiple policies.

Attachment Points

Policies define a set of rules that can be used for various purposes by various applications. Once policies have been created, they need to be applied to take effect. Attachment points describe the specific applications and processes to which policies can be applied.

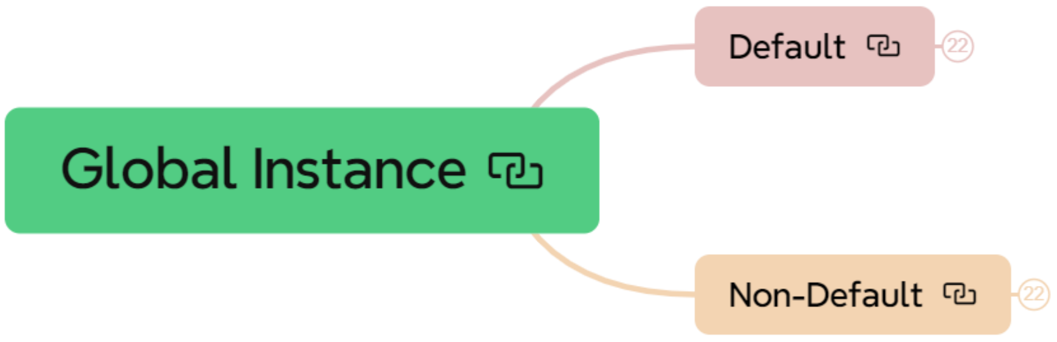
The following diagram shows the BGP attachment points.



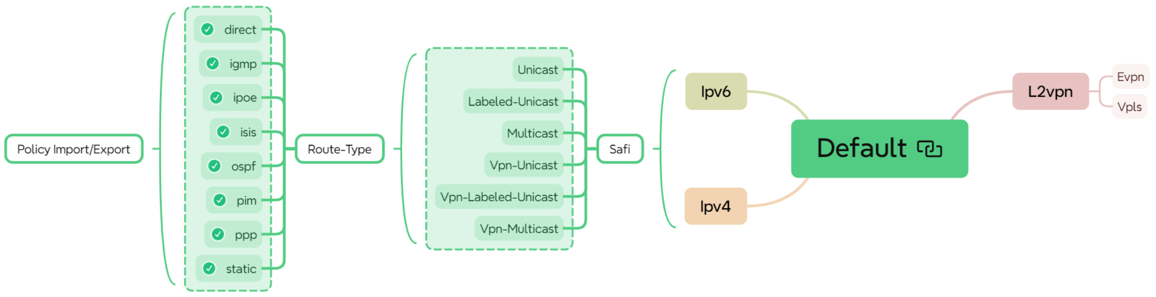
RBFS currently supports the following policy attachment points:

- Instance import/export - Policies attached to an instance at the address family level allow to control which routes will be imported into the instance and exported from the instance by BGP. Such import and export policies are commonly used with BGP L3VPNs.

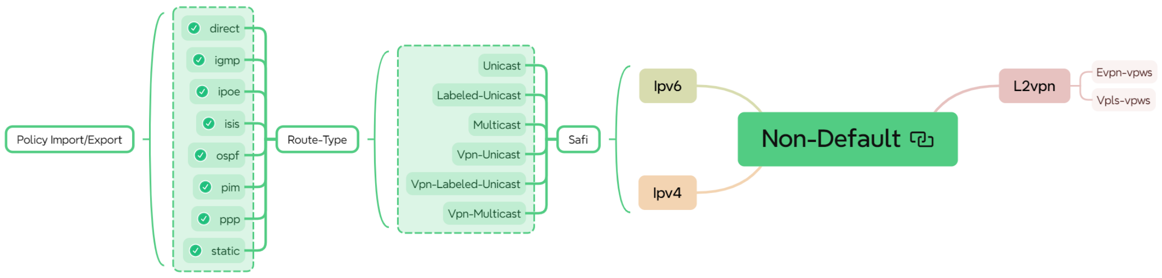
The following diagram shows the BGP Global Instance attachment points.



The following diagram shows the BGP Global Instance default attachment points.

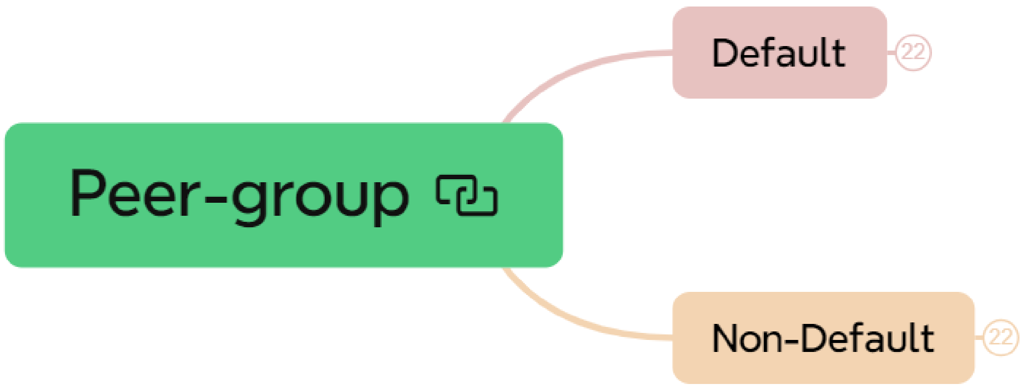


The following diagram shows the BGP Global Instance non-default attachment points.

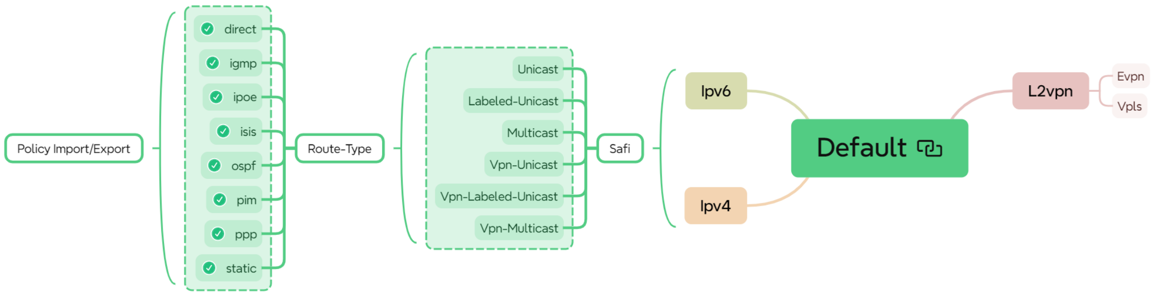


- BGP peer group import/export - Policies attached to BGP peer groups allow to define which routes will be advertised to and accepted from BGP peers. You can attach policies to both instances or peer groups to define the import and export behavior. You can also combine both attachment points, for example if some policy rules apply generically to the instance and some other rules specifically to a peer or peer group.

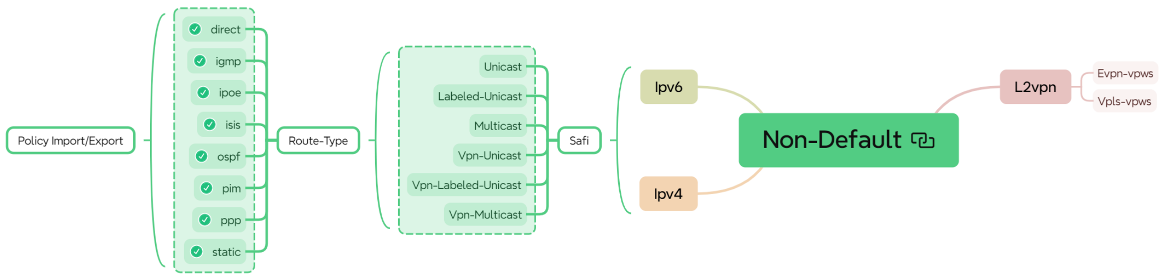
The following diagram shows BGP peer group attachment points.



The following diagram shows BGP peer group default attachment points.

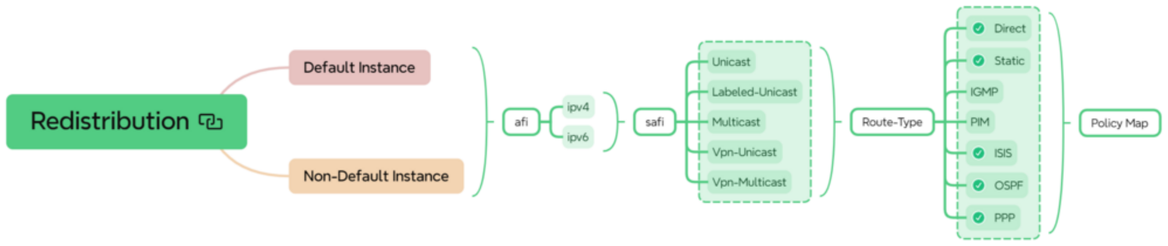


The following diagram shows BGP peer group non-default attachment points.



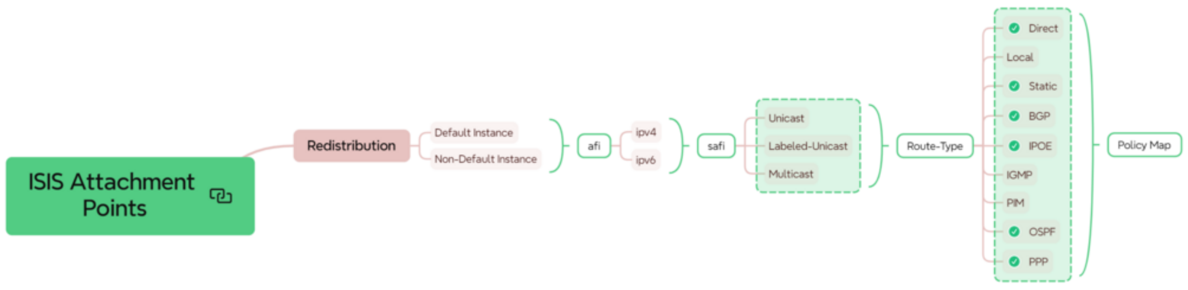
- BGP redistribution - You can attach policies to BGP redistribution to define which routes will be redistributed into the BGP process. This is useful if you would like to redistribute only a sub-set of a type of routes.

The following diagram shows BGP redistribution attachment points.



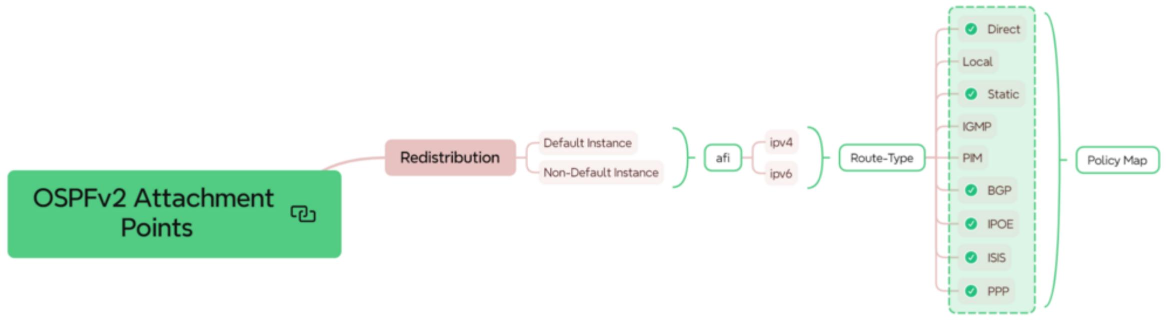
- IS-IS redistribution - Policies can be attached to IS-IS redistribution to control which routes will be redistributed into the IS-IS process.

The following diagram shows IS-IS Redistribution attachment point.



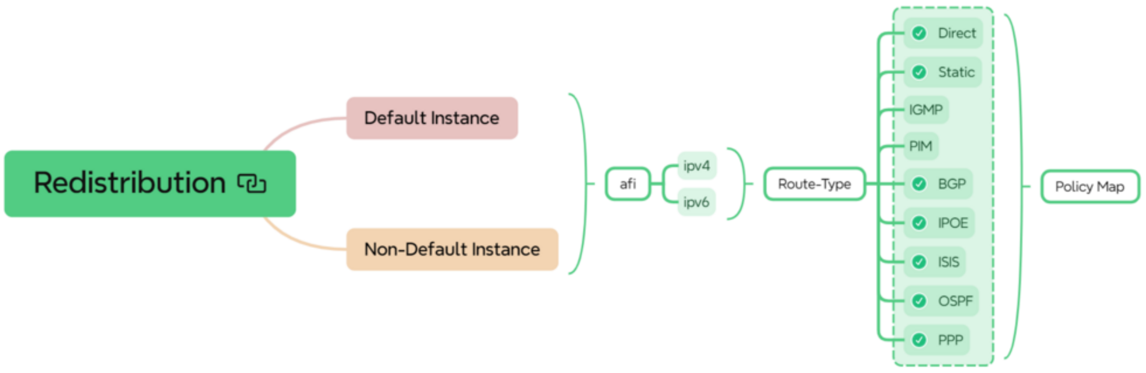
- OSPFv2 redistribution - Policies can be attached to OSPFv2 redistribution to control which routes will be redistributed into the OSPFv2 process.

The following diagram shows OSPF Redistribution attachment points.

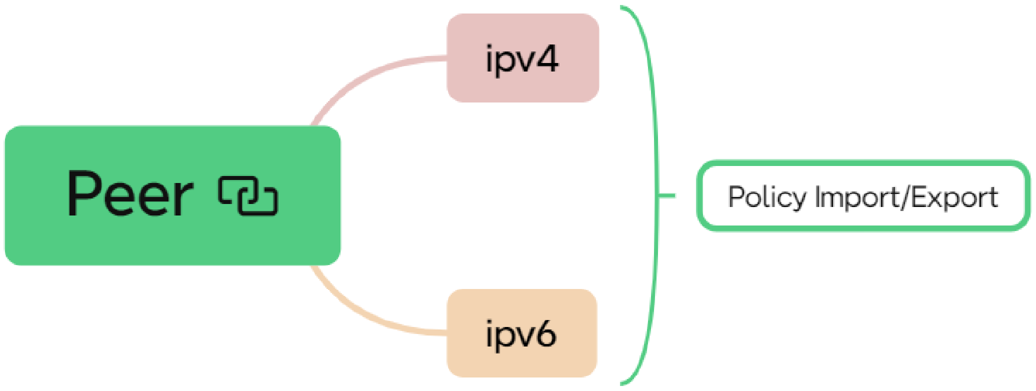


- LDP redistribution - Policies can be attached to the LDP redistribution to control which routes will be redistributed into the LDP process.

The following diagram shows LDP Redistribution attachment points.

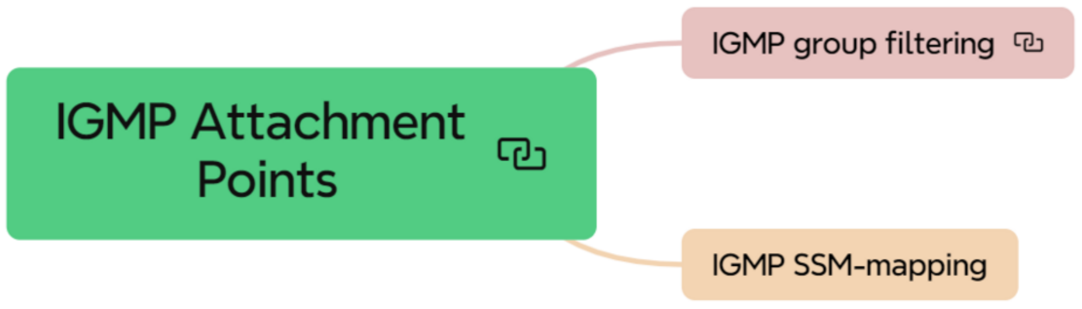


The following diagram shows LDP peer group attachment points.



- IGMP group filtering - You can apply policies to IGMP interface profiles. Such policies act as IGMP group filters when receiving IGMP Membership Report messages.
- IGMP SSM-mapping - Policies are further used to define SSM mapping. SSM mapping policies attached to IGMP interface profiles define how to translate IGMPv2 (*,G) to IGMPv3 (S,G) Reports.

The following diagram shows IGMP attachment points.



Policy Behaviour

The default behaviour of a policy is deny. This means, if a route or any other object is subject to a policy, by default it will be marked as deny. For example in case of an import policy, it will not be imported. A policy will permit an object, for example import a route, if the route or object successfully matches the match rules, and if there is an action rule with a permit. In addition, further operations like modifying route attributes might be executed.

Policy ordinals are executed in the order of the ordinal numbers.

- If an ordinal results in a terminating action like permit or deny, the policy processing is completed for the respective object. Subsequent ordinals will not be processed.
- If an ordinal does not result in a match, the next ordinal is processed.

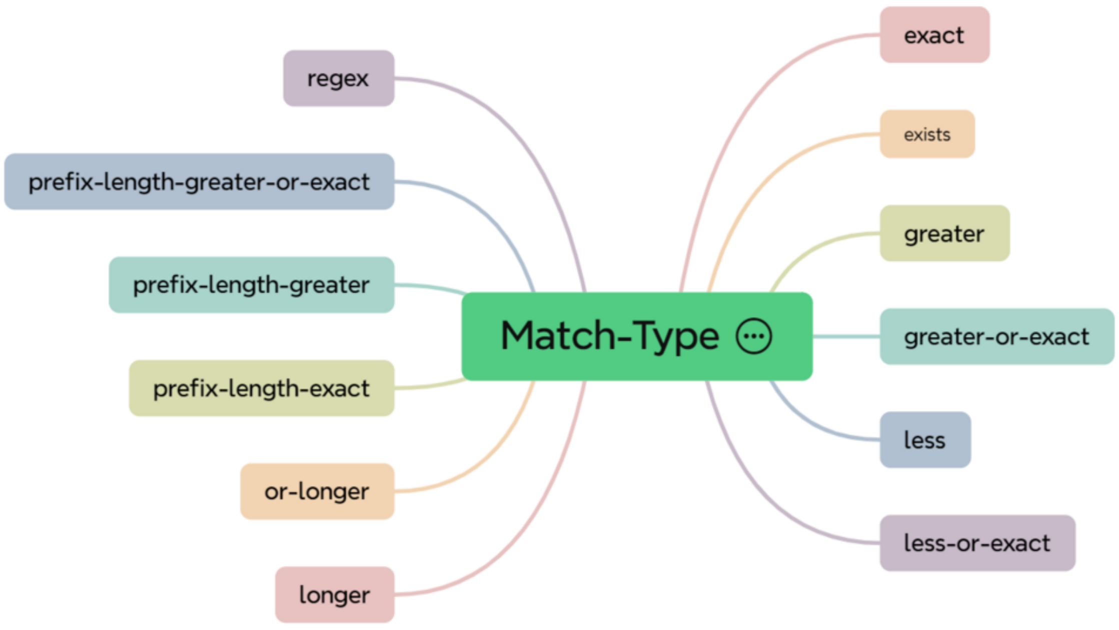
There might be situations in which a policy configuration is not complete or not valid. In particular, a policy may contain a match or action type not supported by an attachment point. For example, the ipv4-mcast-group type is not supported by BGP or IS-IS. The following list summarizes the behaviour for such invalid scenarios:

- If a policy is attached but does not exist, all routes or objects will be denied.
- If a policy contains only statements, or only statements and ordinals, but no match and action block, it will deny all.
- If a match or action type is not supported by the attachment point, the policy will ignore the unsupported rule and process only the supported rules. For any ignored rule, the default deny is not impacted. The behaviour will be the same as if the unsupported rule does not exist.

Match Type

The Match Type refers to the criteria or conditions used to determine if a particular route or set of routes should be acted upon by the policy. It specifies which attributes of the routes are to be evaluated to see if they meet the policy conditions.

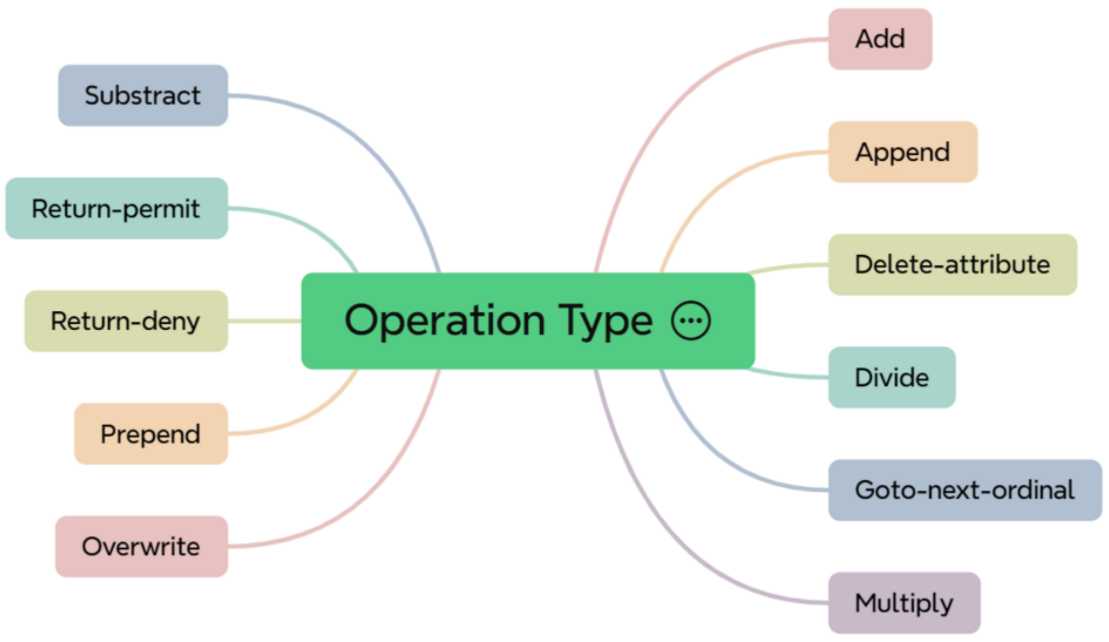
The following diagram shows various match types supported.



Operation Type

The Operation Type refers to the actions applied to the routes that match the specified criteria or match types within the policy. Once a route matches the conditions set by the match types, the operation type dictates what is to be done with the route.

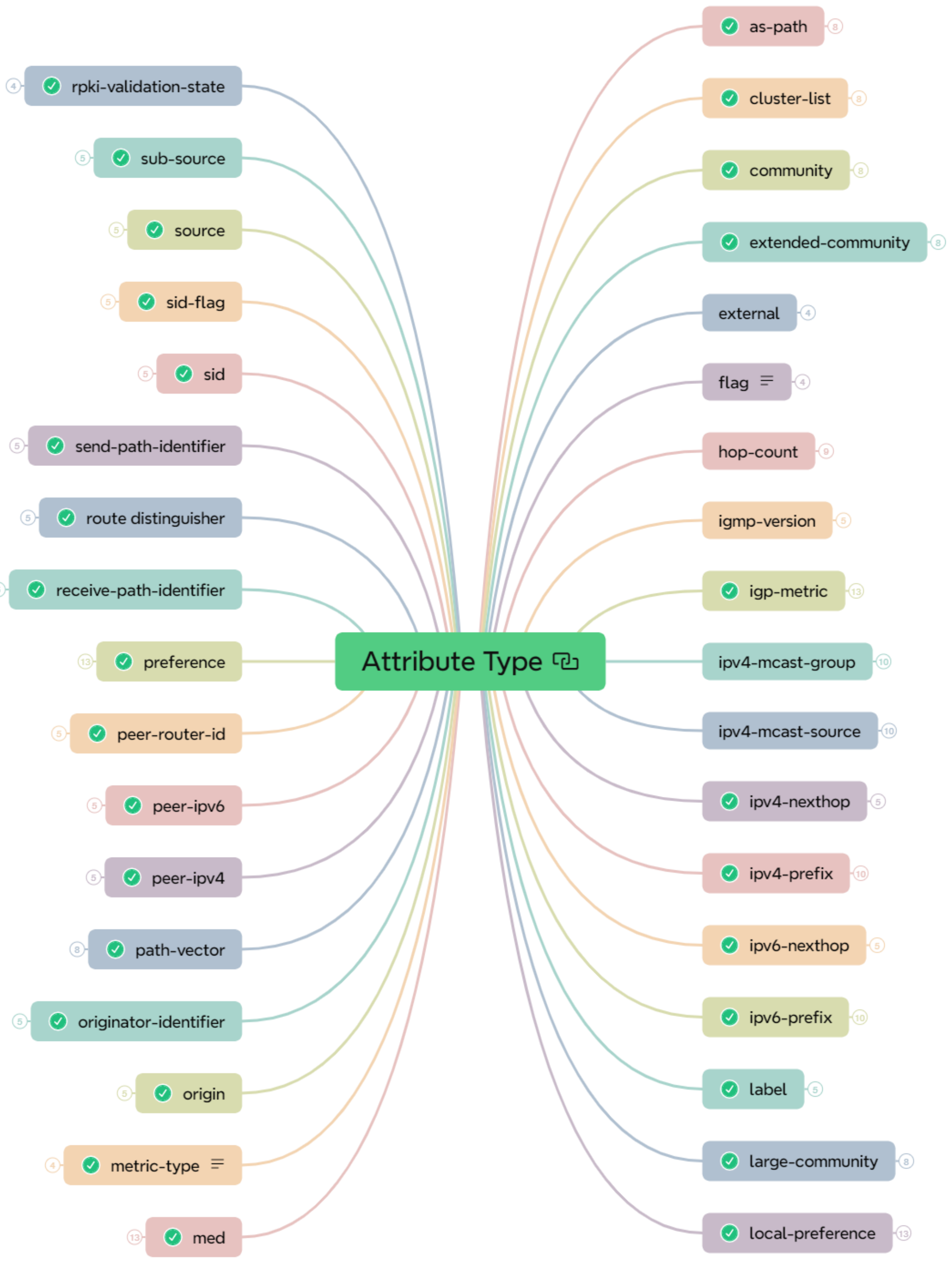
The following diagram shows various operation types supported.



Attribute Type

The Attribute Type refers to the specific properties of routes that can be matched against or modified within the policy. These attributes are used to define conditions (match types) or actions (operation types) on the routes.

The following diagram shows various attribute types supported.



Policy Chaining

Multiservice edge nodes, such as Broadband Network Gateways (BNG) and peering points, manage routes from various sources. The complexity of policy configurations for these routes can be significant. Certain policies, like filtering private IP addresses and removing private Autonomous System (AS) numbers from the AS path, are frequently reused across different router application points.

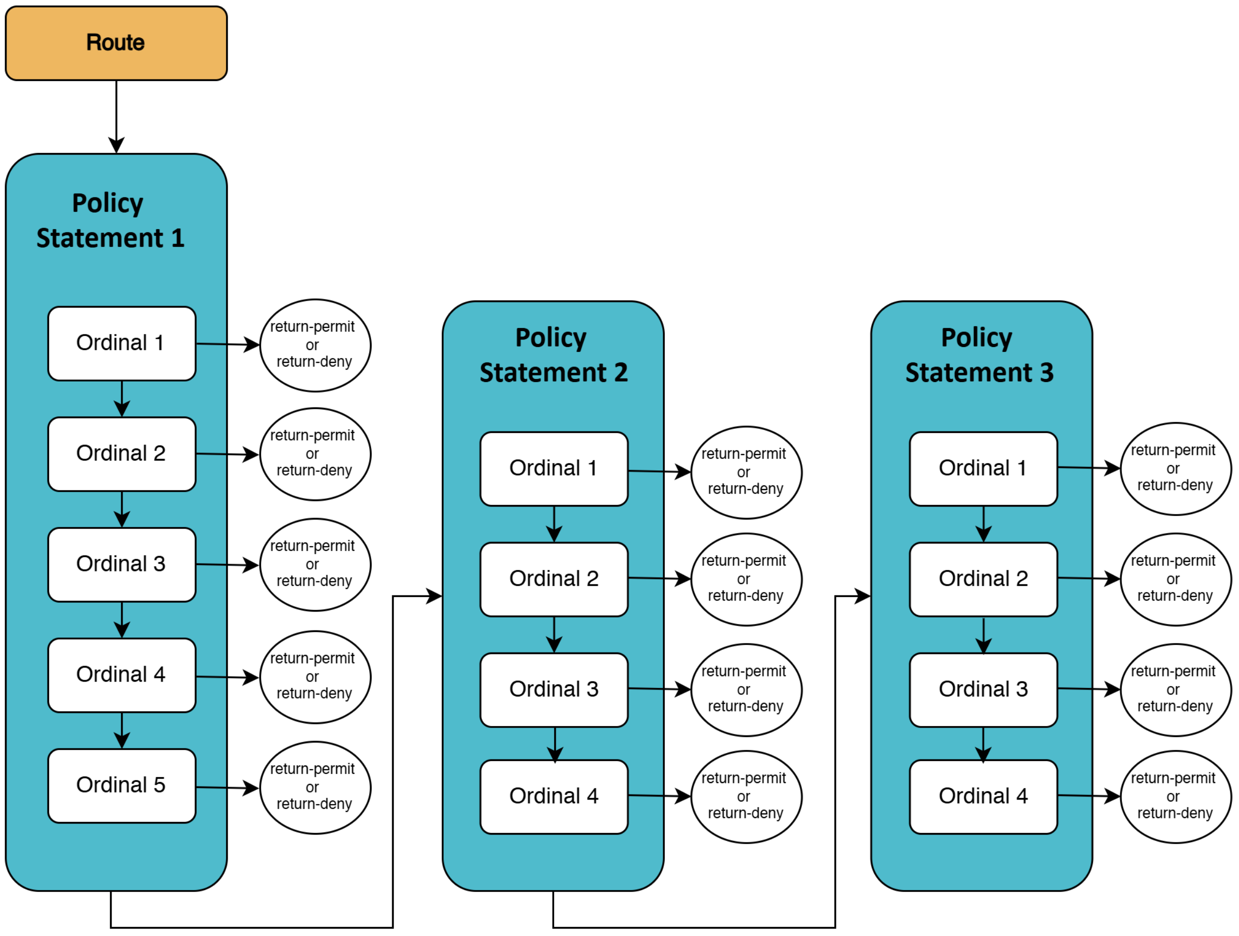
Therefore, implementing modularization and enabling the chaining of these policies improve both maintainability and clarity in policy configurations.

Policy chaining is used to modularize monolithic complex routing policies into multiple smaller policies that are executed in a predefined order and are easier to maintain.

Benefits of using policy chaining:

- **Rule Reusability**: Reuse common rules (for example, RFC1918, bogon filtering) across multiple chains, simplifying configuration.
- **Custom Policies**: Create customer-specific policies by combining existing rules for different peering scenarios (transit, upstream, public).
- **Easier Rule Updates**: Simplify updates to general rules affecting multiple policies.
- **Simplify Troubleshooting**: Make troubleshooting easier by breaking down complex configurations into reusable components.

The diagram below shows how a chain of routing policies is evaluated. When Policy Statements 1-3 are chained in configuration, the actual policy evaluation should proceed as shown in the figure below.



- The route is evaluated starting with the first ordinal in the first routing policy. If there is a match, the specified action is performed.

  If the action is to return-permit or return-deny the route, that action is performed, and the evaluation of the route ends.

  If the goto-next-ordinal action is specified or the route does not match, the evaluation continues by processing the second ordinal in the first routing policy.

  If there is no match in the first ordinal of the first routing policy, the evaluation continues by processing the next ordinal an so on.

  If the route does not match any of the ordinals in the first routing policy, the evaluation continues by processing the ordinals of the second routing policy.

  If the action “goto-next-policy” is specified, the evaluation continues by processing the ordinals of the next policy.

- The evaluation continues until the route matches an ordinal with an action other than goto-next-policy or goto-next-ordinal, or until there are no remaining routing policies to evaluate.

If there are no more routing policies to evaluate, the default action is return-deny.

Support for BGP Community-based Route Leaking Policy

RBFS supports BGP community, extended-community, and large-community as match conditions in route leaking policies. These community attributes can be used to define more flexible route leaking policies.

The table below lists the newly added attributes and match types of route leaking policy.

Attribute Type	Match Types Supported
community	regex exact exists
extended-community	regex exact exists
large-community	regex exact exists

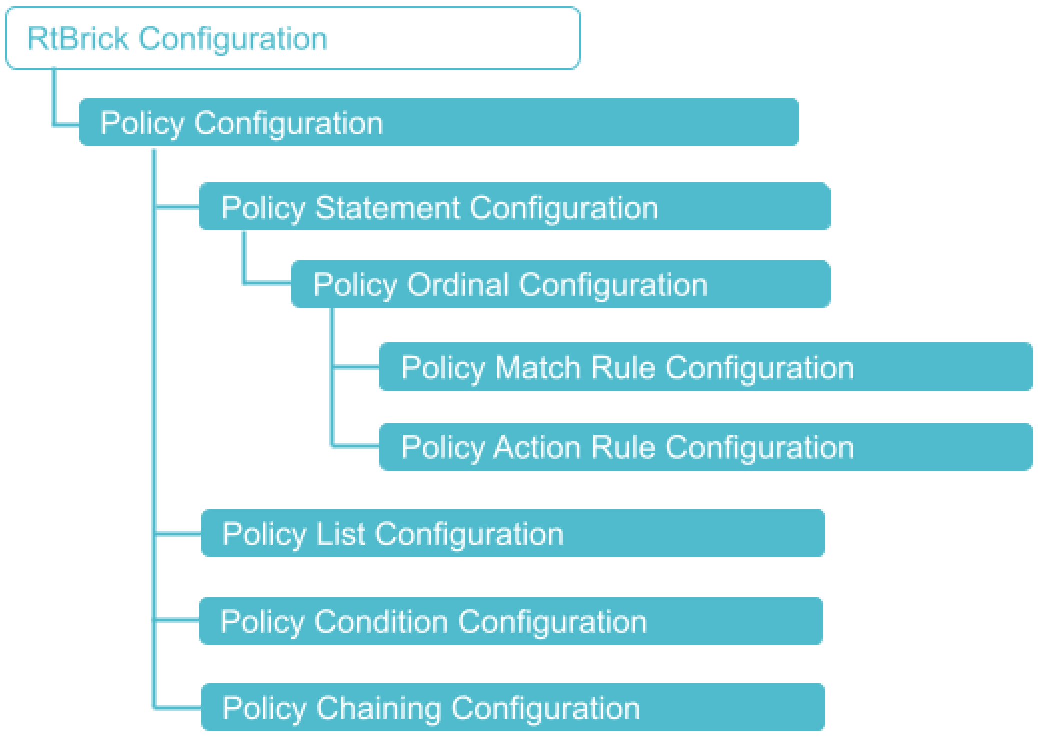
Supported Platforms

Not all features are necessarily supported on each hardware platform. Please refer to the RBFS Platform Guide for the features and the sub-features supported or not supported for each platform.

2.9.2. Policy Configuration

Configuration Hierarchy

The diagram illustrates the policy configuration hierarchy.


RtBrick Configuration
Policy Configuration
Policy Statement Configuration
Policy Ordinal Configuration
Policy Match Rule Configuration
Policy Action Rule Configuration
Policy List Configuration
Policy Condition Configuration
Policy Chaining Configuration

Configuration Syntax and Commands

The following sections describe the policy configuration syntax and commands.

Configuring Policy Statements

Syntax:

set policy statement <policy-name> **ordinal** <number> <attribute> <value>

Attribute	Description
<policy-name>	Name of the policy statement. Policy names can contain alphanumeric characters and underscore character. They must not include special characters like hyphen. For example, **BGP-EXPORT** is not supported, whereas **BGP_EXPORT** is supported. A valid name cannot start with a number but it can contain numbers and underscore in the string. The length of the names should not exceed 64 characters.
<number>	Specifies the ordinal number.
description <text>	Description of the ordinal.

Attribute	Description
match <...>	Match configuration hierarchy. Please refer to section 2.2.1.1 for the match rule configuration.
match-logic <value>	Specifies the match logic. Supported values are and and or.
action <...>	Action configuration hierarchy. Please refer to section 2.2.1.2 for the action rule configuration.
condition <condition-name>	Optionally, apply a policy condition. Please refer to section 2.2.3 for the policy condition configuration.

Configuring Match Rules

Syntax:

set policy statement <policy-name> **ordinal** <number> **match rule** <number> <attribute> <value>

Attribute	Description
rule <number>	Specifies the match rule number.
type <attribute-type>	Specifies the attribute type. Please refer to section 2.2.1.1.1 for supported attribute types.
match-type <match-type>	Specifies the match type. Please refer to section 2.2.1.1.1 for supported match types per attribute, and to section 2.2.1.1.2 for descriptions of the match types.
value <value>	Attribute value. This is the actual value of the attribute to match, for example an IP prefix, a metric, or a community.
value-type <value-type>	Attribute value type. Supported types are discrete, this is a single value, and list, a list of values defined in a policy list.

Attribute and Match Types

Attribute Type	Match Types Supported
ipv4-prefix	regex exact longer or-longer prefix-length-exact prefix-length-greater prefix-length-greater-or-exact
ipv6-prefix	regex exact longer or-longer prefix-length-exact prefix-length-greater prefix-length-greater-or-exact
route-distinguisher	regex exact
community	regex exact exists
extended-community	regex exact exists
large-community	regex exact exists
as-path	regex exact exists
cluster-list	regex exact exists
source	regex exact

Attribute Type	Match Types Supported
sub-source	regex exact
originator-identifier	regex exact
peer-router-id	regex exact
ipv4-nexthop	regex exact
ipv6-nexthop	regex exact
label	regex exact
peer-ipv4	regex exact
peer-ipv6	regex exact
sid	regex exact
sid-flag	regex exact
external	exact
igp-metric	regex exact greater greater-or-exact less less-or-exact
rpki-validation-state	exact

Match Types

Match Types	Description
regex	An attribute can be matched using a standard Linux egrep regular expression. Example: "label": "label-op:push,label:206,bos:1" In this example, the label is a 64bit number, which has label value, bos, and operation encoding. A regex is used to match the string which is displayed in the table dump, that is, label-op:push,label:206,bos:1 not the 64bit value. The same is applicable to an array type attribute. A regex can be written to the string which is visible in the table dump output. For information about Well-known Communities match with Regex, see section "Well-known Communities Regex Match".
exact	Value configured in the command must be same as application attribute value
exists	This is applicable only for array type attribute; an exist match is the one where value configured in the command must exist in the application attribute value which is an array.
less	The application attribute value must be less than the value configured in the command
less-or-exact	The application attribute value must be less than or exact value configured in the command
greater	The application attribute value must be greater than the value configured in the command
greater-or-exact	The application attribute value must be greater than or exact value configured in the command
greater-longer	The route shares the same most-significant bits (described by prefix-length), and prefix-length is greater than the route’s prefix length
greater-or-longer	The route shares the same most-significant bits (described by prefix-length), and prefix-length is equal to or greater than the route’s prefix length.

Match Types	Description
longer	The route address shares the same most-significant bits as the match prefix (destination-prefix or source-prefix). The number of significant bits is described by the prefix-length component of the match prefix. The match will be performed only for prefixes longer than the one supplied for matching.
or-longer	The route address shares the same most-significant bits as the match prefix (destination-prefix or the source-prefix). The number of significant bits is described by the prefix-length component of the match prefix. The match is performed for the prefix supplied and for any prefixes that are subnets of it.
prefix-length-exact	The application attribute value whose prefix length must be lesser than or exact the value configured in the command
prefix-length-greater	The application attribute value whose prefix length must be greater than the value configured in the command
prefix-length-greater-or-exact	The application attribute value whose prefix length must be greater than or exact value configured in the command

Well-known Communities Regex Match

You must specify the value differently to match well-known communities using regex. For example, you can use the following regex pattern to match well-known communities: NO_EXPORT, NO_ADVERTISE, NO_EXPORT_SUBCONFED, and NOOPER.

This pattern works as follows:

NO* matches any string that starts with NO.

This will successfully match the following well-known communities because they start with NO.

- NO_EXPORT
- NO_ADVERTISE
- "NO_EXPORT_SUBCONFED"
- "NOPEER"

Configuring Action Rules

Syntax:

set policy statement <policy-name> **ordinal** <number> **action rule** <number> <attribute> <value>

Attribute	Description
rule <number>	Specifies the action rule number.
operation <operation-type>	Specifies the operation type. Please refer to section 2.2.1.2.1 for supported operations, and to section 2.2.1.2.2 for operations per attribute.
type <attribute-type>	Specifies the attribute type. Please refer to section 2.2.1.2.2 for supported attribute types.
value <value>	Specifies the operation value.

Operation Types

Operation Type	Description
add	The application attribute value will be added with the value configured in the command
append	The application attribute value will be appended with the value configured in the command
delete-attribute	Deletes the attribute from the route. It clears all the information for that specific attribute in the object.
delete-attribute-match	Deletes the specific value of an attribute from the route. It is mandatory to define match rules, before configuring the delete-attribute-match action rule.
divide	The application attribute value will be divided with the value configured in the command
goto-next-ordinal	If next term exists, then next term is executed and the policy result is decided based on the result of the execution
multiply	The application attribute value will be multiplied with the value configured in the command

Operation Type	Description
overwrite	The application attribute value will be overwritten with the value configured in the command
prepend	The application attribute value will be prefixed with the value configured in the command
return-deny	Stops policy execution and returns result as deny (resulting route/BDS object to be denied)
return-permit	Stops policy execution and return result as permit (resulting route/BDS object to be permitted)
subtract	The application attribute value will be subtracted with the value configured in the command. If the result of the subtraction results in a number less than 0, the value "0" is used.

Attribute Types and Supported Operations

Attribute Type	Operation Types Supported
ipv4-prefix	overwrite
ipv6-prefix	overwrite
route-distinguisher	overwrite
community	append prepend overwrite
extended-community	append prepend overwrite
large-community	append prepend overwrite
as-path	append prepend overwrite

Attribute Type	Operation Types Supported
cluster-list	append prepend overwrite
source	overwrite
sub-source	overwrite
originator-identifier	overwrite
peer-router-id	overwrite
ipv4-nexthop	overwrite
ipv6-nexthop	overwrite
label	overwrite
peer-ipv4	overwrite
peer-ipv6	overwrite
sid	overwrite
sid-flag	overwrite
external	overwrite
igp-metric	add subtract multiply divide overwrite
metric-type	overwrite
rpki-validation-state	overwrite

Example 1: Policy statement configuration

```
{
    "rtbrick-config:policy": {
      "statement": [
        {
          "name": "EXPORT_POLICY1",
          "ordinal": [
            {
              "ordinal": 10,
              "description": "Add BGP community to direct routes",
              "match-logic": "and",
              "match": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "ipv6-prefix",
                    "value-type": "discrete",
                    "match-type": "or-longer",
                    "value": "2001:db8:0:60::/32"
                  },
                  {
                    "rule": 2,
                    "type": "source",
                    "value-type": "discrete",
                    "match-type": "exact",
                    "value": "direct"
                  }
                ]
              },
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "community",
                    "operation": "append",
                    "value": "100:1"
                  },
                  {
                    "rule": 2,
                    "operation": "return-permit"
                  }
                ]
              }
            },
            {
              "ordinal": 20,
              "description": "Allow any other route",
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "operation": "return-permit"
                  }
                ]
              }
            }
          ]
        }
      ]
    }
  }
```

Example 2: Policy statement configuration for RPKI Validation State Overwrite

```
{
  "rtbrick-config:policy": {
    "statement": [
      {
        "name": "S1",
        "ordinal": [
          {
            "ordinal": 1,
            "match-logic": "and",
            "match": {
              "rule": [
                {
                  "rule": 1,
                  "type": "rpki-validation-state",
                  "value-type": "discrete",
                  "match-type": "exact",
                  "value": [
                    "invalid"
                    ]
                },
                {
                  "rule": 2,
                  "type": "ipv4-prefix",
                  "value-type": "discrete",
                  "match-type": "exact",
                  "value": [
                    "192.168.0.11/32"
                    ]
                }
              ]
            },
            "action": {
              "rule": [
                {
                  "rule": 1,
                  "type": "rpki-validation-state",
                  "operation": "overwrite",
                  "value": [
                    "unknown"
                    ]
                }
              ]
            }
          }
        ]
      }
    ]
  }
}
```

Configuring Policy Lists

Syntax:

set policy list <list-name> <list-type> **ordinal** <ordinal-number> **value** <value>

Attribute	Description
<list-name>	Name of the policy list

Attribute	Description
<list-type>	Type of the policy list. The following types of lists are supported: - as-path - cluster-list - community - route-distinguisher - extended-community - ipv4-address - ipv4-mcast-group - ipv4-prefix - ipv6-address - ipv6-prefix - large-community - mpls-label - source - sub-source
<ordinal-number>	The number of the list entry.
<value>	The value of the list entry, for example an IP prefix, or a community.

Example: Policy list configuration

```
{
    "rtbrick-config:policy": {
      "list": [
        {
          "name": "PREFIX_LIST1",
          "type": "ipv6-prefix",
          "ordinal": [
            {
              "ordinal": 1,
              "value": "2001:db8:0:10::/32"
            },
            {
              "ordinal": 2,
              "value": "2001:db8:0:25::/32"
            },
            {
              "ordinal": 3,
              "value": "2001:db8:0:30::/32"
            }
          ]
```

```
            }
          ]
        }
    }
```

Configuring Policy Conditions

Policy conditions refer to certain states of the system represented in BDS tables. You need to specify the table, the daemon (BD) that needs to resolve the condition, and if applicable, the attributes used to define the condition.

There are two types of conditions:

- Match on certain attributes in BDS table objects.
- Match on the number of objects in a BDS table.

Configuring Table Match

Syntax:

set policy condition <condition-name> **table** <attribute> <value>

Attribute	**Description**
<condition-name>	Name of the policy condition
name <table-name>	Name of the BDS table
bd <bd-name>	Name of the BD which resolves the condition. Currently supported BDs are: ifmd, ribd, mribd, pppoed, subscriberd, ipoed, l2tpd, pimd, igmp.iod, isis.iod, ospf.appd ospf.iod
count <number>	Optionally, match on the number of objects in a table
match <type>	Type of match for an object count. Supported match types are: equal greater greater-or-equal less less-or-equal

Configuring Attribute Match

You can configure attributes to define a condition. The attributes refer to objects in the table configured in the section above. Please note:

- You can match on multiple attributes.
- In order to identify matching objects, you need to specify all attributes which are primary keys in the table.
- Attribute configuration is not required for conditions matching on the number of table objects using the count option.

Syntax:

set policy condition <condition-name> **attribute** <name> <attribute> <value>

Attribute	Description
<condition-name>	Name of the policy condition
attribute <name>	Name of the attribute in the BDS table
value <value>	Value of the attribute to match
match <type>	Type of the match. Supported match types are: equal exists greater greater-or-equal less less-or-equal regex

Example: Policy condition configuration

```
{
    "rtbrick-config:policy": {
      "condition": [
        {
          "condition_name": "precheck",
          "table": {
            "name": "global.instance",
            "bd": "bgp.iod.1"
          },
          "attribute": [
            {
              "name": "instance_name",
              "match": "equal",
              "value": "default"
            }
          ]
        }
      ]
    }
  }
```

Attaching Policies

Once a policy has been created, they need to be applied to an application like a routing protocol to take effect.

BGP Attachment Points

- Instance import/export
- BGP peer group import/export
- BGP redistribution

For attaching policies to the BGP protocol, please refer to the RBFS BGP User Guide.

IS-IS Attachment Points

- IS-IS redistribution

For attaching policies to the IS-IS protocol, please refer to the IS-IS User Guide.

OSPFv2 Attachment Points

- OSPFv2 redistribution

For attaching policies to the OSPFv2/v3 protocol, please refer to the OSPFv2 User Guide.

LDP Attachment Points

- LDP redistribution#

For attaching policies to the LDP protocol, please refer to the LDP User Guide.

IGMP Attachment Points

- IGMP group filtering
- IGMP SSM-mapping

For attaching policies to the IGMP protocol, please refer to the RBFS IP Multicast Routing Configuration Guide.

Policy Chaining Configuration

The following configuration demonstrates how policy chaining is applied for the PEERING_V4 BGP peer group. For IPv4 unicast address family, incoming routes will be evaluated and imported only if they meet the criteria defined in the associated import policies. These policies provide layered route filtering, ensuring that only valid and desired routes are accepted by the peer.

The configuration applies a series of import policies for the IPv4 unicast address family. These policies are chained in sequence, each performing specific tasks such as:

- Filtering bogon IPs (FILTER_BOGONS_V4).
- Preventing the import of the peer’s own IP ranges (FILTER_OWN_V4).
- Scrubbing unwanted community tags (SCRUB_COMMUNITIES).

The following example shows the policy chaining configuration for the peer-group “PEERING_V4.”

```
set instance default protocol bgp peer-group PEERING_V4
set instance default protocol bgp peer-group PEERING_V4 remote-as 65536
set instance default protocol bgp peer-group PEERING_V4 address-family ipv4
labeled-unicast
set instance default protocol bgp peer-group PEERING_V4 address-family ipv4
unicast
set instance default protocol bgp peer-group PEERING_V4 address-family ipv4
unicast policy import FILTER_BOGONS_V4
set instance default protocol bgp peer-group PEERING_V4 address-family ipv4
unicast policy import FILTER_OWN_V4
set instance default protocol bgp peer-group PEERING_V4 address-family ipv4
unicast policy import SCRUB_COMMUNITIES
```

```
supervisor@rtbrick>SPINE01: cfg> show config instance default protocol bgp peer-
group PEERING_V4
{
  "rtbrick-config:peer-group": [
    {
      "pg-name": "PEERING_V4",
      "remote-as": 65536,
      "address-family": [
        {
          "afi": "ipv4",
          "safi": "labeled-unicast"
        },
        {
          "afi": "ipv4",
          "safi": "unicast",
          "policy": {
            "import": [
              "FILTER_BOGONS_V4",
              "FILTER_OWN_V4",
              "SCRUB_COMMUNITIES"              ]
          }
        },
        {
          "afi": "ipv6",
          "safi": "labeled-unicast"
        },
        {
          "afi": "ipv6",
          "safi": "unicast"
        }
      ]
    }
  ]
}
supervisor@rtbrick>SPINE01: cfg>
```

Policy Statement Configurations for Policy Chaining

The following configurations provide a detailed representation of the policy statements utilized in the aforementioned policy chaining example.

The example below shows the policy definitions for the FILTER_BOGONS_V4 policy statement.

```
set policy statement FILTER_BOGONS_V4
set policy statement FILTER_BOGONS_V4 ordinal 1
set policy statement FILTER_BOGONS_V4 ordinal 1 description FILTER_BOGONS
set policy statement FILTER_BOGONS_V4 ordinal 1 match-logic or
set policy statement FILTER_BOGONS_V4 ordinal 1 match
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 1
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 1 type ipv4-prefix
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 1 value-type list
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 1 match-type or-longer
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 1 value FILTER_BOGONS
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 2
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 2 type ipv4-prefix
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 2 value-type discrete
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 2 match-type regex
set policy statement FILTER_BOGONS_V4 ordinal 1 match rule 2 value 0.0.0.0/(2[5-
9]|3[0-1])
set policy statement FILTER_BOGONS_V4 ordinal 1 action
set policy statement FILTER_BOGONS_V4 ordinal 1 action rule 1
set policy statement FILTER_BOGONS_V4 ordinal 1 action rule 1 operation return-
deny
set policy statement FILTER_BOGONS_V4 ordinal 2
set policy statement FILTER_BOGONS_V4 ordinal 2 description FILTER_IXP
set policy statement FILTER_BOGONS_V4 ordinal 2 match
set policy statement FILTER_BOGONS_V4 ordinal 2 match rule 1
set policy statement FILTER_BOGONS_V4 ordinal 2 match rule 1 type ipv4-prefix
set policy statement FILTER_BOGONS_V4 ordinal 2 match rule 1 value-type list
set policy statement FILTER_BOGONS_V4 ordinal 2 match rule 1 match-type or-longer
set policy statement FILTER_BOGONS_V4 ordinal 2 match rule 1 value
IXP_PEERING_LANS_V4
set policy statement FILTER_BOGONS_V4 ordinal 2 action
set policy statement FILTER_BOGONS_V4 ordinal 2 action rule 1
set policy statement FILTER_BOGONS_V4 ordinal 2 action rule 1 operation return-
deny
```

The example below shows the configurations of the ‘FILTER_BOGONS’ policy list used in the ‘FILTER_BOGONS_V4’ policy statement.

```
supervisor@S1-STD-2-2004>bm04-tst.fsn.rtbrick.net: op> show config set policy list
FILTER_BOGONS
set policy list FILTER_BOGONS ipv4-prefix
set policy list FILTER_BOGONS ipv4-prefix ordinal 1
set policy list FILTER_BOGONS ipv4-prefix ordinal 1 value 0.0.0.0/8
set policy list FILTER_BOGONS ipv4-prefix ordinal 2
set policy list FILTER_BOGONS ipv4-prefix ordinal 2 value 10.0.0.0/8
set policy list FILTER_BOGONS ipv4-prefix ordinal 3
set policy list FILTER_BOGONS ipv4-prefix ordinal 3 value 100.64.0.0/10
set policy list FILTER_BOGONS ipv4-prefix ordinal 4
set policy list FILTER_BOGONS ipv4-prefix ordinal 4 value 127.0.0.0/8
```

The example below shows the configurations of the ‘IXP_PEERING_LANS_V4’ policy list used in the ‘FILTER_BOGONS_V4’ policy statement.

```
set policy list IXP_PEERING_LANS_V4 ipv4-prefix
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 1
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 1 value 80.81.192.0/21
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 2
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 2 value 80.249.208.0/21
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 3
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 3 value 91.213.211.0/24
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 4
set policy list IXP_PEERING_LANS_V4 ipv4-prefix ordinal 4 value 185.0.20.0/23
```

The example below shows the policy definitions for the FILTER_OWN_V4 policy statement.

```
set policy statement FILTER_OWN_V4
set policy statement FILTER_OWN_V4 ordinal 1
set policy statement FILTER_OWN_V4 ordinal 1 match
set policy statement FILTER_OWN_V4 ordinal 1 match rule 1
set policy statement FILTER_OWN_V4 ordinal 1 match rule 1 type ipv4-prefix
set policy statement FILTER_OWN_V4 ordinal 1 match rule 1 value-type list
set policy statement FILTER_OWN_V4 ordinal 1 match rule 1 match-type or-longer
set policy statement FILTER_OWN_V4 ordinal 1 match rule 1 value MYPREFIXES_V4
set policy statement FILTER_OWN_V4 ordinal 1 action
set policy statement FILTER_OWN_V4 ordinal 1 action rule 1
set policy statement FILTER_OWN_V4 ordinal 1 action rule 1 operation return-deny
```

The example below shows the configurations of the ‘MYPREFIXES_V4’ policy list used in the ‘FILTER_OWN_V4’ policy statement.

```
set policy list MYPREFIXES_V4 ipv4-prefix
set policy list MYPREFIXES_V4 ipv4-prefix ordinal 1
set policy list MYPREFIXES_V4 ipv4-prefix ordinal 1 value 192.0.2.0/24
set policy list MYPREFIXES_V4 ipv4-prefix ordinal 2
set policy list MYPREFIXES_V4 ipv4-prefix ordinal 2 value 198.51.100.0/24
set policy list MYPREFIXES_V4 ipv4-prefix ordinal 3
set policy list MYPREFIXES_V4 ipv4-prefix ordinal 3 value 203.0.113.0/24
```

The example below shows the policy definitions for the SCRUB_COMMUNITIES policy statement.

```
set policy statement SCRUB_COMMUNITIES
set policy statement SCRUB_COMMUNITIES ordinal 1
set policy statement SCRUB_COMMUNITIES ordinal 1 match
set policy statement SCRUB_COMMUNITIES ordinal 1 match rule 1
set policy statement SCRUB_COMMUNITIES ordinal 1 match rule 1 type community
set policy statement SCRUB_COMMUNITIES ordinal 1 match rule 1 value-type discrete
set policy statement SCRUB_COMMUNITIES ordinal 1 match rule 1 match-type regex
set policy statement SCRUB_COMMUNITIES ordinal 1 match rule 1 value 64496:*
set policy statement SCRUB_COMMUNITIES ordinal 1 action
set policy statement SCRUB_COMMUNITIES ordinal 1 action rule 1
set policy statement SCRUB_COMMUNITIES ordinal 1 action rule 1 type community
set policy statement SCRUB_COMMUNITIES ordinal 1 action rule 1 operation delete-
attribute-match
```

Sample Configurations

Example 1: BGP export policy referencing a policy list

```
supervisor@leaf1: cfg> show config policy
{
  "rtbrick-config:policy": {
    "list": [
      {
        "name": "PREFIX_LIST2",
        "type": "ipv6-prefix",
        "ordinal": [
          {
            "ordinal": 1,
            "value": "2001:db8:0:60::/32"
          },
          {
            "ordinal": 2,
            "value": "2001:db8:0:80::/64"
          },
          {
            "ordinal": 3,
            "value": "2001:db8:0:110::/64 "
          }
        ]
      }
    ],
    "statement": [
      {
        "name": "EXPORT_POLICY2",
        "ordinal": [
          {
            "ordinal": 10,
            "description": "Add community to direct routes",
            "match": {
              "rule": [
                {
                  "rule": 1,
                  "type": "source",
                  "value-type": "discrete",
                  "match-type": "exact",
                  "value": "direct"
                }
              ]
            },
            "action": {
              "rule": [
                {
                  "rule": 1,
                  "type": "community",
                  "operation": "append",
                  "value": "100:1"
                },
                {
                  "rule": 2,
                  "operation": "return-permit"
                }
              ]
            }
          },
          {
            "ordinal": 20,
            "description": "Allow list of routes",
            "match": {
              "rule": [
                {
                  "rule": 1,
                  "type": "ipv6-prefix",
                  "value-type": "list",
                  "match-type": "or-longer",
                  "value": "PREFIX_LIST2"
                }
              ]
            },
            "action": {
              "rule": [
                {
                  "rule": 1,
                  "operation": "return-permit"
                }
              ]
            }
          },
          {
            "ordinal": 30,
            "description": "Deny any other route",
            "action": {
              "rule": [
                {
                  "rule": 1,
                  "operation": "return-deny"
                }
              ]
            }
          }
        ]
      }
    ]
  }
}
```

```
supervisor@leaf1: cfg> show config instance default protocol bgp peer-group spine
address-family ipv6 unicast
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv6",
      "safi": "unicast",
      "policy": {
        "export": "EXPORT_POLICY2"
      }
    }
  ]
}
```

Example 2: IGMP filter policy

```
supervisor@leaf1: cfg> show config policy
{
  "rtbrick-config:policy": {
    "statement": [
      {
        "name": "IGMP_FILTER",
        "ordinal": [
          {
            "ordinal": 1,
            "description": "IGMP group filter",
            "match-logic": "or",
            "match": {
              "rule": [
                {
                  "rule": 1,
                  "type": "ipv4-mcast-group",
                  "value-type": "discrete",
                  "match-type": "or-longer",
                  "value": "198.51.100.0/24"
                },
                {
                  "rule": 2,
                  "type": "ipv4-mcast-group",
                  "value-type": "discrete",
                  "match-type": "or-longer",
                  "value": "198.51.100.0/24"
                }
              ]
            },
            "action": {
              "rule": [
                {
                  "rule": 1,
                  "operation": "return-permit"
                }
              ]
            }
          }
        ]
      }
    ]
  }
}
```

```
supervisor@leaf1: cfg> show config multicast-options igmp
{
  "rtbrick-config:igmp": {
    "interface-profile": [
      {
        "profile-name": "PROFILE1",
        "filter-policy": "IGMP_FILTER"
      }
    ]
  }
}
```

Example 4: Community Deletion Configuration

```
supervisor@rtbrick: cfg> show config policy
{
  "rtbrick-config:policy": {
    "statement": [
      {
        "name": "test_policy",
        "ordinal": [
          {
            "ordinal": 1,
            "match-logic": "or",
            "match": {
              "rule": [
                {
                  "rule": 1,
                  "type": "community",
                  "value-type": "discrete",
                  "match-type": "exists",
                  "value": [
                    "1:1"
                    ]
                },
                {
                  "rule": 2,
                  "type": "extended-community",
                  "value-type": "discrete",
                  "match-type": "exists",
                  "value": [
                    "target:3.3.3.3:3000",
                    "target:5.5.5.5:5000"
                    ]
                },
                {
                  "rule": 3,
                  "type": "large-community",
                  "value-type": "discrete",
                  "match-type": "exists",
                  "value": [
                    "3333:33333:33",
                    "4444:44444:44"
                    ]
                },
                {
                  "rule": 4,
                  "type": "community",
                  "value-type": "discrete",
                  "match-type": "regex",
                  "value": [
                    "555:*"
                    ]
                }
              ]
            },
            "action": {
              "rule": [
                {
                  "rule": 1,
                  "type": "community",
                  "operation": "delete-attribute-match"
                },
                {
                  "rule": 2,
                  "type": "extended-community",
                  "operation": "delete-attribute-match"
                },
                {
                  "rule": 3,
                  "type": "large-community",
                  "operation": "delete-attribute-match"
                }
              ]
            }
          }
        ]
      }
    ]
  }
```

```
}
```

2.9.3. Policy Operational Commands

Policy Test

You can use the policy test feature to test a policy before attaching it to a protocol or an instance.

Perform the following tasks:

- Step 1: Identify the brick daemon that will process the policy and the table to which the policy will be applied.
- Step 2: Execute the 'test policy run' command.

Example: Testing a BGP VPN export policy

```
supervisor@leaf1: op> test policy run bgp.appd.1 policy-name VPN_V4_EXPORT table
default.bgp.rib-in.import.ipv4.vpn-unicast
```

- Step 3: View the test results.

The policy test feature will create two result tables. The result table ending with ".policy.permit" will show all objects permitted by the policy, the one ending with ".policy.deny" will show all objects denied by the policy.

Example: Viewing the result tables

```
supervisor@leaf1: op> show datastore bgp.appd.1 table default.bgp.rib-
in.import.ipv4.vpn-unicast.policy.permit
<...>
supervisor@leaf1: op> show datastore bgp.appd.1 table default.bgp.rib-
in.import.ipv4.vpn-unicast.policy.deny
<...>
```

- Step 4: Clear the result tables

You can clear the result tables using the 'test policy clear' command. Apply the clear command to the same table for which you have run the policy test.

Example: Clearing the result tables

```
supervisor@leaf1: op> test policy clear bgp.appd.1 policy-name VPN_V4_EXPORT table
default.bgp.rib-in.import.ipv4.vpn-unicast
```

2.10. Static Routing

2.10.1. Static Routing Overview

Static routing allows a network administrator to configure routes manually. Using the RtBrick CLI, you can configure static IPv4, IPv6, MPLS, and multicast routes.

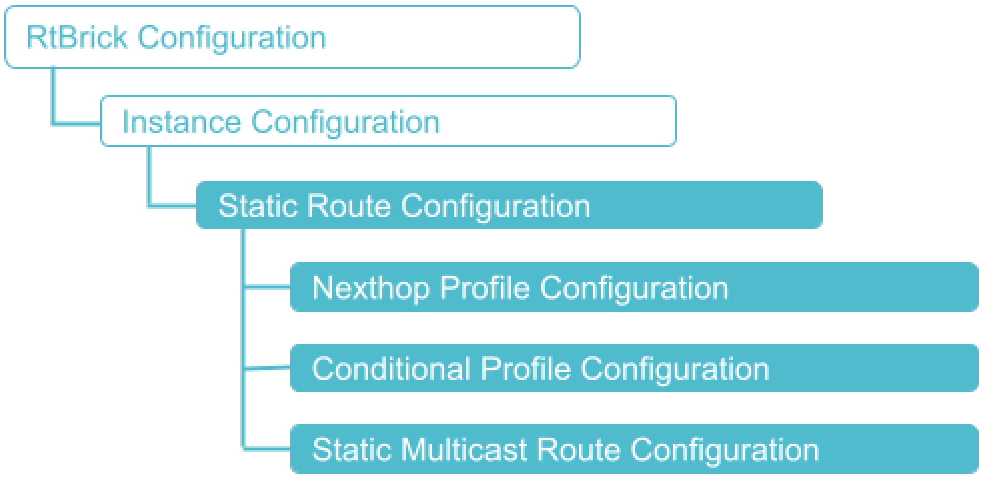
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the Platform Guide for the features and the sub-features that are or are not supported by each platform.

2.10.2. Static Routing Configuration

Configuration Hierarchy

The diagram illustrates the static routes configuration hierarchy.


RtBrick Configuration
Instance Configuration
Static Route Configuration
Nexthop Profile Configuration
Conditional Profile Configuration
Static Multicast Route Configuration

Configuration Syntax and Commands

The following sections describe the static route configuration syntax and commands. In RBFS, next hops of static routes are configured separately, and referenced by the actual routes.

Static Route Configuration

This section describes how to configure the static route itself.

Syntax

set instance <instance-name> **static route** <attribute> <value>

Attribute	Description
<instance-name>	Name of the routing instance
<afi> <prefix | label> (true | false)	Supported AFIs are ipv4, ipv6, and mpls. In case of IPv4 or IPv6, configure the prefix. In case of MPLS, configure the incoming label and BOS flag.
<safi>	Supported SAFIs are unicast, labeled-unicast, and multicast.
<nexthop-profile>	Name of the nexthop profile
route-options no-install	This option keeps the routes in the routing information base, but prevents them from being used for forwarding traffic.

Example: Static Route Configuration

```
{
    "rtbrick-config:route": {
      "ipv4": [
        {
          "prefix4": "198.51.100.15/24",
          "safi": "unicast",
          "nexthop-profile": "nexthop1",
          "preference": 20

        }
      ],
      "ipv6": [
        {
          "prefix6": "2001:db8:0:117::/32",
          "safi": "unicast",
          "nexthop-profile": "nexthop2"
        }
      ],
      "mpls": [
        {
          "in-label": 8888,
          "in-bos": "true",
          "safi": "unicast",
          "nexthop-profile": "nexthop1"
        }
      ]
    }
  }
```

Example: Static Route Configuration with route-options no install

```
supervisor@rtbrick.net: cfg> show config instance default static route ipv4
192.1.0.6/32
{
  "rtbrick-config:ipv4": [
    {
      "prefix4": "192.1.0.6/32",
      "safi": "labeled-unicast",
      "nexthop-profile": "nh1",
      "route-options": "no-install"
    },
    {
      "prefix4": "192.1.0.6/32",
      "safi": "unicast",
      "nexthop-profile": "nh1"
    }
  ]
}
```

Nexthop Profile Configuration

You can group various nexthop parameters with a nexthop profile name and instance, and associate this nexthop profile with multiple routes.

Syntax

set instance <instance-name> **static nexthop-profile** <name> <attribute> <value>

Attribute	Description
<instance-name>	Name of the routing instance
nexthop-profile <name>	Nexthop profile name
exit-interface <exit-interface>	Exit interface name
lookup-afi (ipv4 | ipv6 | mpls)	Lookup routing table address family where the nexthop will be resolved.
lookup-instance <lookup-instance>	Lookup routing table instance where the nexthop will be resolved.

Attribute	Description
lookup-safi (labeled-unicast | multicast | unicast)	Lookup routing table subsequent address family where the nexthop will be resolved.
nexthop <address>	IPv4/IPv6 nexthop address
out-bos (true | false)	Label BOS
out-label <out-label>	Label to be pushed
resolve-direct true	The option restricts all routes from resolving the nexthop of a static route instead, it allows only the direct routes to resolve the nexthop of a static route.

Example: Nexthop Profile Configuration

```
{
    "rtbrick-config:static": {
      "nexthop-profile": [
        {
          "name": "nexthop1",
          "nexthop": "198.51.100.145",
          "out-label": 4444
        },
        {
          "name": "nexthop3",
          "exit-interface": "ifp-0/0/4/4"
        }
      ]
    }
  }
```

- If you do not provide lookup-instance, lookup-afi and lookup-safi values, default values will be used to install the route.
- The exit interface attribute is mandatory for link-local nexthop.

Conditional Profile Configuration

By using the conditional static route feature, you can make specific routes conditional. These conditional routes are installed only if the specified condition is satisfied.

You can group various conditional parameters such as match-instance, match-afi, match-safi, compare-operation, compare-type, and compare-value with a conditional profile name, and associate this conditional profile with multiple routes.

Syntax:

set instance <instance-name> **static conditional-profile** <name> <attribute> <value>

Attribute	Description
conditional <name>	Conditional profile name
compare-operation greater-then	Conditional routing compare operation
compare-type route-count	Conditional routing compare type
compare-value <compare-value>	Conditional routing condition value
match-instance <instance-name>	Routing instance where the condition will be checked.
match-afi (ipv4 |ipv6 |mpls)	Routing tables address family (AFI) for which the condition will be checked.
match-safi (labeled-unicast |multicast |unicast)	Routing table subsequent address family (SAFI) for which the condition will be checked.

Example: Conditional Profile Configuration

```
{
    "rtbrick-config:conditional-profile": [
      {
        "name": "c2",
        "match-instance": "default",
        "match-afi": "ipv4",
        "match-safi": "unicast",
        "compare-type": "route-count",
        "compare-operation": "greater-than",
        "compare-value": 20
      }
    ]
  }
```

Static Multicast Route Configuration

Syntax:

set instance <instance-name> **static route multicast4** <attribute> <value

Attribute	Description
<instance-name>	Name of the routing instance
<source>	IPv4 multicast source address
<group>	IPv4 multicast group address

Example: Static Multicast Route Configuration

```
{
    "rtbrick-config:static": {
      "route": {
        "multicast4": [
          {
            "source": "198.51.100.15/24",
            "group": "198.51.100.35/24",
            "nexthop-profile": "nexthop3"
          }
        ]
      }
    }
}
```

2.10.3. Static Routing Operational Commands

Show Commands

Static Routes Created by staticd

These commands show static routes as created by the static route daemon (staticd).

Syntax:

show static route <options>

Attribute	Description
<afi>	Supported AFIs are ipv4, ipv6, and mpls.
<safi>	Supported SAFIs are unicast, labeled-unicast, and multicast.
instance <name>	Static routes for an instance

Example: List static routes for all instances

```
supervisor@dev1: cfg> show static route
Instance: default, AFI: ipv4, SAFI: unicast
Prefix/Label              Pref    Next Hop                  Interface
198.51.100.100/24          2       198.51.100.22                   -
Instance: default, AFI: ipv6, SAFI: unicast
Prefix/Label              Pref    Next Hop                  Interface
2001:db8:0:334::/32                2     2001:db8:0:99::                      -
```

Static Routes in the Routing Table

These commands show the static routes included in the final routing table.

Syntax:

show route <options> **source static** <options>

Attribute	Description
<afi>	Supported AFIs are ipv4, ipv6, and mpls.
<safi>	Supported SAFIs are unicast, labeled-unicast, and multicast.
detail	Detailed route information
instance <name>	Routing table information for a specific instance
label <value>	Destination label
mpls	Address family
prefix <value>	Destination prefix
source	Source of the routing information

Example 1: List static routes information

```
supervisor@dev1: cfg> show route ipv4 source static
Instance: default, AFI: ipv4, SAFI: unicast
Prefix/Label                                Source           Pref    Next Hop
Interface
198.51.100.100/24                            static           2       198.51.100.22
ifl-0/0/1/4
supervisor@dev1: cfg>
```

Example 2: List detailed static routes information

```
supervisor@dev1: cfg> show route ipv4 source static detail
Instance: default, AFI: ipv4, SAFI: unicast
198.51.100.100/24
  Source: static, Preference: 2
    Next Hop: 198.51.100.22
      Covering prefix: 198.51.100.22/24
      Next Hop type: direct, Next Hop action: None
      Resolved in: default-ipv4-unicast
      Egress interface: ifl-0/0/1/4, NextHop MAC: 7a:00:81:64:04:04
```

Example 3: List MPLS route information

```
supervisor@rtbrick: cfg> show route mpls
Instance: default, AFI: mpls, SAFI: unicast
Prefix/Label    Source            Pref    Next Hop                Interface
20010           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20011           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20012           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20013           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20014           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20015           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20016           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20017           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
20018           bgp               170     2001:db8:0:110::        ifl-0/0/17/1001
```

3. Layer 2 Services

3.1. L2X

3.1.1. L2X Overview

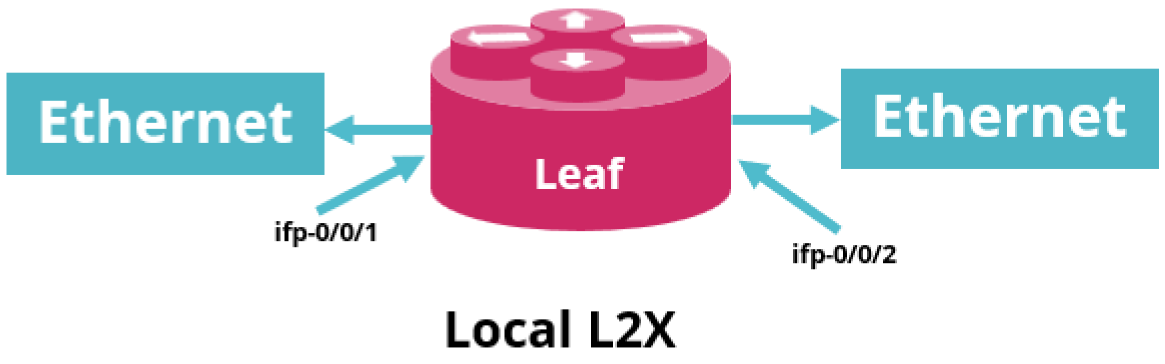
Layer 2 Cross-Connect (L2X) is a data plane feature that connects two physical ports (IFPs) using Layer 2 switching. L2X can switch the traffic between two IFPs to provide the trunk service for an Ethernet switch.

Local and Remote L2X

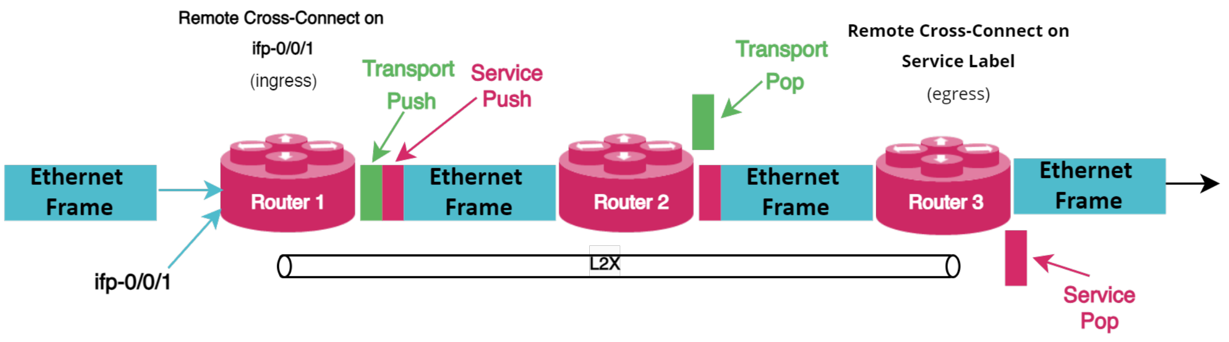
Local L2X refers to an L2 connection between two ports or VLANs on the same device. In a local L2X, both interfaces are on the same router. The L2X can switch Layer 2 (frame) traffic between the ports. Based on the configuration, these cross connects can be uni-directional as well as bi-directional.

Remote L2X refers to L2 connection between two ports located on two different devices. In a remote L2X, the interfaces are located on two different routers and it requires an MPLS tunnel to transport the traffic between the two routers.

Local L2X: The following figure shows the Local L2X scenario.



Remote L2X: The following figure shows the remote L2X scenario.



Unidirectional and Bidirectional L2X

Unidirectional refers to data either sent or received in one direction and Bidirectional implies the flow of traffic between two routers in both directions.

The bidirectional cross connect feature helps you to establish cross connection between two local ports with an L2X configuration. Bi-directional attribute is applicable only to local cross connect. Bidirectional connectivity requires a pair of unidirectional L2X or a single bidirectional L2X.

> The VLAN operations are not supported for bi-directional local cross-connect.

Ingress and Egress in L2X

In L2X, ingress traffic is incoming traffic that enters the boundary of a network and egress traffic implies outgoing traffic that exits an entity or a network boundary.

Port and VLAN Cross-connects

Both port and VLAN cross-connects switches Layer 2 traffic from input interface to output interface. A port cross-connect switches all Layer 2 traffic arriving at an input interface, but a VLAN cross-connect only switches the Layer 2 traffic associated with a specific VLAN. A port-based L2X indicates a port-only configuration, so there are no VLANs involved.

Both single-tag and double-tagged (inner and outer VLAN tags) are supported. The port and VLAN L2X support both local and remote L2X configurations. In remote L2X connections, the VLAN cross-connects are typically configured on the MPLS tunnel ingress router.

Untagged traffic on L2X interfaces is also supported. However, there is no way to select only untagged traffic for cross-connecting. Therefore, only port cross connects are supported for untagged traffic.

L2X 802.1ad Ethertype Support

RBFS supports VLAN operations such as VLAN add, VLAN swap, and VLAN delete on egress interface. RBFS supports similar functionality at the ingress side as well. That is, RBFS supports the following VLAN operations:

- Single-VLAN-Add with an option to configure encapsulation (that is, 802.1q or 802.1ad)
- Single-VLAN-Delete
- Swap-Outer-VLAN

By default the encapsulation method is 802.1q. If an encapsulation method is not specified, 802.1q is the default mode.

In addition to setting the Ethertype for a VLAN operation, the 802.1ad support includes that ingress traffic for all tagged match options will match on both Ethertype 0x8100 (802.1q) and 0x88a8 (802.1ad) by default.

VLAN Operations

RBFS supports VLAN operations such as VLAN add, VLAN swap and VLAN delete on Ingress and Egress interfaces.

The current functionality has been extended to all the existing CLIs to accept ingress and egress VLAN operations and Ingress and Egress VLAN encapsulation values.

Both 802.1q and 802.1ad encapsulations are supported. The default encapsulation is 802.1q.

Traffic will be matched at ingress direction based on the match criterion. RtBrick Full Stack (RBFS) supports the following match parameters.

On a physical interface, there are five different match types. Traffic can be matched based on the following:

1. (ifp)

2. (ifp, outer_vlan)
3. (ifp, outer_vlan, inner_vlan)
4. (ifp, outer_vlan, any inner_vlan)
5. (ifp, any vlan)

Some of the match types are mutually exclusive. For example, (ifp, outer_vlan, inner_vlan) and (ifp, outer_vlan, any inner_vlan) configuration on the same interface throws errors.

If ifp, any vlan match type is configured with any other match type, it will create conflicts.

The match-type attribute is mandatory for match-untagged, match-any and match-inner-any match criteria.

VLAN Change Operation for IFLs

RBFS allows users to change VLAN configurations for logical interfaces (IFLs) without deleting the existing VLAN configurations. This functionality is applicable to both L2 and L3 IFLs.

Supported Match Type Validations

The following table shows the supported match type validations.

The asterisk * indicates *any* or *no vlan* tags.

Cases	Configuration A	Configuration B	Support
Case 1 : IFP A, *	IFP A,*	IFP A, ov 10	No
	IFP A,*	IFP A, ov 10, iv 20	No
	IFP A,*	IFP A, ov 10, *	No
	IFP A,*	IFP A, untagged	No
Case 2: IFP A, untagged	IFP A, untagged	IFP A, *	No
	IFP A, untagged	IFP A, ov 10	Yes
	IFP A, untagged	IFP A, ov 30, iv 20	Yes
	IFP A, untagged	IFP A, ov 20, *	Yes

Cases	Configuration A	Configuration B	Support
Case 3: IFP A, outer_vlan:	IFP A, ov 10	IFP A, *	No
	IFP A, ov 10	IFP A, ov 10, *	No
	IFP A, ov 10	IFP A, ov 20	Yes
	IFP A, ov 10	IFP A, ov 10 , iv 20	No
	IFP A, ov 10	IFP A, ov 40 , iv 7	Yes
	IFP A, ov 10	IFP A, ov 30, *	Yes
	IFP A, ov 10	IFP A, untagged	Yes
Case 4: IFP A, outer_vlan, inner_vlan:	IFP A, ov 10, iv 20	IFP A, *	No
	IFP A, ov 10, iv 20	IFP A, ov 10, *	No
	IFP A, ov 10, iv 20	IFP A, ov 10	No
	IFP A, ov 10, iv 20	IFP A, ov 30	Yes
	IFP A, ov 10, iv 20	FP A, ov 20 , *	Yes
	IFP A, ov 10, iv 20	IFP A, untagged	Yes
	IFP A, ov 10, iv 20	IFP A, ov 10, iv 30	Yes
Case 5: IFP A, outer_vlan, *	IFP A, ov 10, *	IFP A, *	No
	IFP A, ov 10, *	IFP A, ov 10	No
	IFP A, ov 10, *	IFP A, ov 10, iv 20	No
	IFP A, ov 10, *	IFP A, ov 20, iv 7	Yes
	IFP A, ov 10, *	IFP A, ov 30	Yes
	IFP A, ov 10, *	IFP A, untagged	Yes
	IFP A, ov 10, *	IFP A, ov 40, *	Yes

Match Type Options

The supported match-type option includes:

- match-any: Matches any frame, regardless of VLAN tags.
- match-inner-any: Matches frames based on their inner VLAN tag.
- match-outer: Matches frames based on their outer VLAN tag.
- match-outer-inner: Matches frames based on both outer and inner VLAN tags.

- match-service-label: Matches frames based on a specific MPLS label.
- match-untagged: Matches untagged frames (frames without VLAN tags).

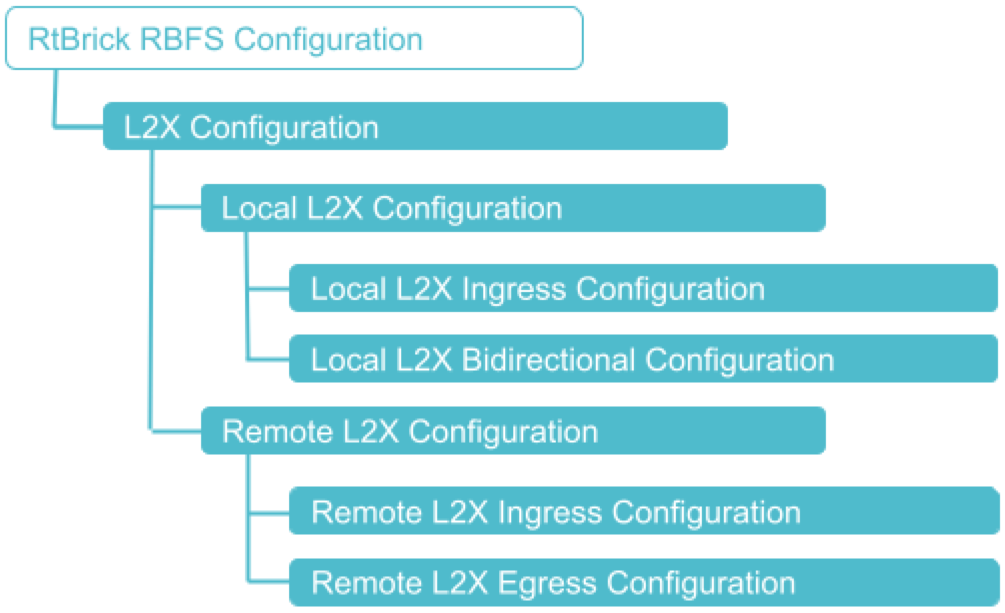
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

3.1.2. L2X Configuration

Configuration Hierarchy

The diagram illustrates the L2X configuration hierarchy.


RtBrick RBFS Configuration
L2X Configuration
Local L2X Configuration
Local L2X Ingress Configuration
Local L2X Bidirectional Configuration
Remote L2X Configuration
Remote L2X Ingress Configuration
Remote L2X Egress Configuration

Configuration Syntax and Commands

The following sections describe the L2X configuration syntax and commands.

Local L2X Configuration

The following sections describe the Local L2X unidirectional and bidirectional configurations.

Local L2X Ingress Configuration

This configuration enables local unidirectional L2X (Local Cross-Connect) on the same device.

Syntax:

set l2x name <l2x-name> **ingress** <attribute> <value>

Attribute	Description
<l2x-name>	Name of L2X
description <description>	(Optional) L2X description
egress-vlan-encapsulation <encapsulation>	(Optional) Egress VLAN encapsulation value. NOTE: This option can be configured only after ‘swap-outer-vlan’ configuration.
egress-vlan-operation <vlan-action>	(Optional) Outgoing VLAN operation
incoming-inner-vlan <vlan-id>	(Optional) Incoming inner VLAN
incoming-interface <incoming-interface>	(Mandatory) Incoming physical interface name
incoming-outer-vlan <vlan-id>	(Optional) Incoming outer VLAN
ingress-outer-vlan <vlan-id>	(Optional) Outer VLAN at ingress side
ingress-vlan-encapsulation <encapsulation>	(Optional) Ingress VLAN encapsulation value. NOTE: This option can be configured only after ‘swap-outer-vlan’ configuration.
ingress-vlan-operation <vlan-action>	(Optional) VLAN operation on ingress side outer VLAN
match-type <match-type>	(Mandatory) L2X match type
match-tp-id 802.1ad	The option enables L2X to recognize and handle packets tagged with 0x88a8 TPID, allowing interoperability with 802.1ad encapsulated traffic.

Attribute	Description
outgoing-interface <outgoing-interface>	(Mandatory) Outgoing physical interface name
outgoing-outer-vlan <vlan-id>	(Optional) Outgoing outer VLAN

Example 1: Local L2X Ingress Configuration with Port Match

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test1",
          "direction": "ingress",
          "incoming-interface": "ifp-0/1/64",
          "outgoing-interface": "ifp-0/1/66",
          "match-type": "match-any"
        }
      ]
    }
  }
```

Example 2: Local L2X Ingress Configuration with VLAN Match

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test4",
          "direction": "ingress",
          "incoming-interface": "ifp-0/1/12",
          "incoming-outer-vlan": 200,
          "outgoing-interface": "ifp-0/1/13",
          "match-type": "match-outer"
        }
      ]
    }
  }
```

Example 3: Local L2X Ingress VLAN Operations

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test4",
          "direction": "ingress",
          "incoming-interface": "ifp-0/1/12",
          "incoming-outer-vlan": 200,
          "outgoing-interface": "ifp-0/1/13",
          "match-type": "match-outer",
          "ingress-vlan-operation": "single-vlan-add",
          "ingress-outer-vlan": 400
        }
      ]
    }
  }
```

Example 4: Local L2X Configuration with 0x88a8 TPID tagged packets.

```
"rtbrick-config:l2x": {
      "name": [
        {
          "name": "local_1",
          "direction": "ingress",
          "incoming-interface": "ifp-0/0/3",
          "incoming-outer-vlan": 100,
          "outgoing-interface": "ifp-0/0/22",
          "egress-vlan-operation": "swap-outer-vlan",
          "outgoing-outer-vlan": 100,
          "egress-vlan-encapsulation": "802.1ad",
          "match-type": "match-outer",
          "ingress-vlan-operation": "single-vlan-delete",
          "match-tp-id": "802.1ad"
        }
      ]
    },
```

Local L2X Bidirectional Configuration

This configuration enables redirecting traffic incoming (ingress) on a particular interface to another interface and vice versa on the same hardware device.

Syntax:

set l2x name <l2x-name> **bi-directional** <attribute> <value>

Attribute	Description
match-type <match-type>	(Mandatory) Match types with which traffic can be matched.
incoming-interface <incoming-interface>	(Mandatory) Incoming interface is where the traffic originates from.
outgoing-interface <outgoing-interface>	(Mandatory) Outgoing interface where the traffic is going to.
description <description>	(Optional) L2X description

Attribute	Description
egress-vlan-encapsulation <encapsulation>	(Optional) Egress VLAN encapsulation
incoming-inner-vlan <vlan-id>	(Optional) Incoming inner VLAN
incoming-outer-vlan <vlan-id>	(Optional) Incoming outer VLAN
ingress-outer-vlan <vlan-id>	(Optional) Outer VLAN at ingress side
ingress-vlan-encapsulation <encapsulation>	(Optional) Ingress VLAN encapsulation
outgoing-outer-vlan <vlan-id>	(Optional) Outgoing outer VLAN

Example 1: Local L2X Bidirectional Configuration

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test2",
          "direction": "bi-directional",
          "incoming-interface": "ifp-0/1/64",
          "outgoing-interface": "ifp-0/1/66",
          "match-type": "match-untagged"
        }
      ]
    }
  }
```

Local L2X VLAN Operation Configuration

This configuration enables VLAN operations for both ingress and egress traffic on a particular interface to another interface and vice versa on the same hardware device.

Syntax:

set l2x name <l2x-name> **ingress [egress-vlan-operation | ingress-vlan-operation]** <attribute> <value>

Attribute	Description
<l2x-name>	Name of the l2x.
double-vlan-add	Adds two VLAN tags (QinQ tagging) to the packet.
double-vlan-delete	Removes two VLAN tags from the packet.
single-vlan-add	Adds a single VLAN tag to the packet.
single-vlan-delete	Removes a single VLAN tag from the packet.
swap-outer-vlan	Replaces the outer VLAN tag with a new one.

Remote L2X Configuration

The following sections describe the remote L2X configurations.

Remote L2X Ingress Configuration

This configuration enables the remote L2X ingress side.

Syntax:

set l2x name <l2x-name> **ingress** <attribute> <value>

Attribute	Description
match-type <match-type>	(Mandatory) Match types with which traffic can be matched. The 'match-type' option includes match-any, match-inner-any, match-outer, match-outer-inner, match-service-label, and match-untagged. For more details, see the Match Type Options section.
incoming-interface <incoming-interface>	(Mandatory) Incoming interface is where the traffic originates from.
ingress-vlan-operation <ingress-vlan-action>	(Optional) VLAN operation on ingress side outer VLAN
ingress-outer-vlan <vlan-id>	(Optional) Outer VLAN at ingress side
ingress-vlan-encapsulation <encapsulation>	(Optional) Ingress VLAN encapsulation value
nexthop4/nexthop6 <nexthop>	(Mandatory) Next-Hop address

Attribute	Description
lookup-instance <lookup-instance>	(Optional) Instance name
lookup-afi <lookup-afi>	(Optional) AFI value: ipv4 or ipv6
lookup-safi <lookup-safi>	(Optional) SAFI value: safi values are unicast, labeled-unicast
service-label <service_label>	(Mandatory) Service label value. NOTE: Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. It is recommended to assign label values outside of these reserved ranges to avoid conflicts.

Example 1: Remote L2X Ingress Configuration with Port Match

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test8",
          "direction": "ingress",
          "incoming-interface": "ifp-0/1/64",
          "nexthop4": "198.51.100.44",
          "lookup-instance": "default",
          "lookup-afi": "ipv4",
          "lookup-safi": "labeled-unicast",
          "service-label": 10000,
          "match-type": "match-any"
        }
      ]
    }
  }
```

Example 2: Remote L2X Ingress Configuration with VLAN match

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test4",
          "direction": "ingress",
          "incoming-interface": "ifp-0/1/12",
          "incoming-outer-vlan": 100,
          "incoming-inner-vlan": 200,
          "nexthop4": "198.51.100.44",
          "lookup-instance": "default",
          "lookup-afi": "ipv4",
          "lookup-safi": "labeled-unicast",
          "service-label": 8000,
          "match-type": "match-outer-inner"
        }
      ]
    }
  }
```

Example 3: Remote L2X Ingress VLAN operations

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test4",
          "direction": "ingress",
          "incoming-interface": "ifp-0/1/12",
          "incoming-outer-vlan": 100,
          "incoming-inner-vlan": 200,
          "nexthop4": "198.51.100.44",
          "lookup-instance": "default",
          "lookup-afi": "ipv4",
          "lookup-safi": "labeled-unicast",
          "service-label": 10000,
          "match-type": "match-outer-inner",
          "ingress-vlan-operation": "Single-Vlan-Delete"
        }
```

Remote L2X Egress Configuration

This configuration enables the remote L2X egress side.

Syntax:

set l2x name <l2x-name> **egress** <attribute> <value>

Attribute	Description
service-label <service_label>	(Mandatory) Service label value. NOTE: Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range between 20000 - 100000. It is recommended to assign a label value outside of these reserved ranges to avoid conflicts.
outgoing-interface <outgoing-interface>	(Mandatory) Interface where traffic is going to.
egress-vlan-operation <vlan-action>	(Optional) Outgoing VLAN operation

Attribute	Description
outgoing-outer-vlan <vlan-id>	(Optional) Outgoing outer VLAN

Example 1: Local L2X Egress Configuration

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test4",
          "direction": "egress",
          "service-label": 10000,
          "outgoing-interface": "ifp-0/1/66"
        }
      ]
    }
  }
```

Example 2: Local L2X Egress Configuration with VLAN Operation

```
{
    "rtbrick-config:l2x": {
      "name": [
        {
          "name": "test4",
          "direction": "egress",
          "service-label": 10000,
          "outgoing-interface": "ifp-0/1/12",
          "egress-vlan-operation": "single-vlan-add",
          "outgoing-outer-vlan": 400
        }
      ]
    }
  }
```

3.1.3. L2X Operational Commands

The L2X show commands provide detailed information about the L2X operations.

L2X Show Commands

The L2X show commands display data from FIB local table. Therefore, local L2X with down ports or remote l2x with unresolved nexthop address are not displayed.

L2X Summary

The summary commands display L2X information in a tabular format. Key information is displayed in the summary output.

> The L2X name is truncated after certain length as space is less to display summary output. In such cases, you can use detail command output where full name is displayed.

Syntax:

show l2x <options>

Option	Description
-	Without any option, the commands displays all L2X information such as ingress L2X and egress L2X.
l2x-name <l2x-name>	Displays information for a specific L2X.
detail	Displays detailed L2X information for a specific L2X.
direction <direction>	Displays L2X information for a specified direction, where direction can be ingress, egress, or bi-directional.
local-interface <interface-name>	Displays L2X information for a specific LAG interface.
nexthop4 <nexthop>	Displays L2X information for the remote IPv4 address.
nexthop6 <nexthop>	Displays L2X information for the remote IPv6 address.
service-label <service_label>	Displays the L2X information for a specific service label.
type <type>	Displays detailed L2X information for a specific type and L2X.
statistics	Displays statistics for a specific L2X.

Example 1: Summary view of L2X information

```
supervisor@rtbrick: op> show l2x
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop     Outer VLAN
Inner VLAN      Service label
l2bsa-0/1/27/281479271677953   ingress    ifp-0/1/27           2001:db8:0:75::            64
Any          110011
l2bsa-0/1/27/281479271677953   egress      -                   ifp-0/1/27                  -                -
120011
l2bsa-0/1/27/281479271677954   ingress    ifp-0/1/27           2001:db8:0:75::          65                Any
```

```
110012
l2bsa-0/1/27/281479271677954   egress     -                    ifp-0/1/27              -                -
120012
l2bsa-0/1/27/281479271677955   ingress    ifp-0/1/27           2001:db8:0:75::         66               Any
110013
l2bsa-0/1/27/281479271677955   egress     -                    ifp-0/1/27              -                -
120013
l2bsa-0/1/27/281479271677956   ingress    ifp-0/1/27           2001:db8:0:75::         67               Any
110014
l2bsa-0/1/27/281479271677956   egress     -                    ifp-0/1/27              -                -
120014
```

Example 2: Summary view of a specific L2X

```
supervisor@rtbrick: op> show l2x l2bsa-0/1/27/281479271677953
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN
Inner VLAN      Service label
l2bsa-0/1/27/281479271677953   ingress    ifp-0/1/27           2001:db8:0:75::           64               Any
110011
l2bsa-0/1/27/281479271677953   egress     -                    ifp-0/1/27                -                -
120011
```

Example 3: Summary view of a remote L2X

```
supervisor@rtbrick: op> show l2x type remote
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN
Inner VLAN      Service label
l2bsa_lag-1_11                 ingress    lag-1                2001:db8:0:85::           11               Any
120011
l2bsa_lag-1_11                 egress     -                    lag-1                     -                -
110011
l2bsa_lag-1_12                 ingress    lag-1                2001:db8:0:85::           12               Any
120012
l2bsa_lag-1_12                 egress     -                    lag-1                     -                -
110012
l2bsa_lag-1_13                 ingress    lag-1                2001:db8:0:85::           13               Any
120013
l2bsa_lag-1_13                 egress     -                    lag-1                     -                -
110013
l2bsa_lag-1_14                 ingress    lag-1                2001:db8:0:85::           14               Any
120014
l2bsa_lag-1_14                 egress     -                    lag-1                     -                -
110014
l2bsa_lag-1_15                 ingress    lag-1                2001:db8:0:85::           15               Any
120015
l2bsa_lag-1_15                 egress     -                    lag-1                     -                -
110015
l2bsa_lag-1_16                 ingress    lag-1                2001:db8:0:85::           16               Any
120016
```

Example 4: Summary view of L2X for a specific service label

```
supervisor@rtbrick: op> show l2x service-label 120011
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN    Inner
VLAN      Service label
l2bsa_lag-1_11                 ingress    lag-1                2001:db8:0:85::           11            Any
120011
supervisor@rtbrick: op>
```

Example 5: Summary view of the L2X information for a specified direction, where direction can be ingress, egress, or bi-directional.

```
supervisor@rtbrick: op> show l2x direction ingress
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN
Inner VLAN      Service label
l2bsa_lag-1_11                 ingress    lag-1                2001:db8:0:85::           11               Any
120011
l2bsa_lag-1_12                 ingress    lag-1                2001:db8:0:85::           12               Any
120012
l2bsa_lag-1_13                 ingress    lag-1                2001:db8:0:85::           13               Any
120013
l2bsa_lag-1_14                 ingress    lag-1                2001:db8:0:85::           14               Any
120014
l2bsa_lag-1_15                 ingress    lag-1                2001:db8:0:85::           15               Any
120015
l2bsa_lag-1_16                 ingress    lag-1                2001:db8:0:85::           16               Any
120016
l2bsa_lag-1_17                 ingress    lag-1                2001:db8:0:85::           17               Any
120017
l2bsa_lag-1_18                 ingress    lag-1                2001:db8:0:85::           18               Any
120018
l2bsa_lag-1_19                 ingress    lag-1                2001:db8:0:85::           19               Any
120019
l2bsa_lag-1_20                 ingress    lag-1                2001:db8:0:85::           20               Any
120020
l2bsa_lag-1_21                 ingress    lag-1                2001:db8:0:85::           21               Any
120021
l2bsa_lag-1_22                 ingress    lag-1                2001:db8:0:85::           22               Any
120022
l2bsa_lag-1_23                 ingress    lag-1                2001:db8:0:85::           23               Any
120023
l2bsa_lag-1_24                 ingress    lag-1                2001:db8:0:85::           24               Any
120024
l2bsa_lag-1_25                 ingress    lag-1                2001:db8:0:85::           25               Any
120025
l2bsa_lag-1_26                 ingress    lag-1                2001:db8:0:85::           26               Any
120026
l2bsa_lag-1_27                 ingress    lag-1                2001:db8:0:85::           27               Any
120027
```

Example 6: Summary view of L2X information for a specific LAG interface

```
supervisor@rtbrick: op> show l2x local-interface lag-1
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN
Inner VLAN      Service label
l2bsa_lag-1_11                 ingress    lag-1                2001:db8:0:85::            11
Any             120011
l2bsa_lag-1_12                 ingress    lag-1                2001:db8:0:85::            12
Any             120012
l2bsa_lag-1_13                 ingress    lag-1                2001:db8:0:85::            13
Any             120013
l2bsa_lag-1_14                 ingress    lag-1                2001:db8:0:85::            14
Any             120014
l2bsa_lag-1_15                 ingress    lag-1                2001:db8:0:85::            15
Any             120015
l2bsa_lag-1_16                 ingress    lag-1                2001:db8:0:85::            16
Any             120016
l2bsa_lag-1_17                 ingress    lag-1                2001:db8:0:85::            17
Any             120017
l2bsa_lag-1_18                 ingress    lag-1                2001:db8:0:85::            18
Any             120018
l2bsa_lag-1_19                 ingress    lag-1                2001:db8:0:85::            19
Any             120019
l2bsa_lag-1_20                 ingress    lag-1                2001:db8:0:85::            20
Any             120020
```

Example 7: Summary view of L2X information for a remote egress router

```
supervisor@rtbrick: op> show l2x nexthop4 198.51.100.103
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN
```

```
Inner VLAN      Service label
l2bsa_lag-1_11                 ingress    lag-1                198.51.100.103           11               Any
120011
l2bsa_lag-1_12                 ingress    lag-1                198.51.100.103           12               Any
120012
l2bsa_lag-1_13                 ingress    lag-1                198.51.100.103           13               Any
120013
l2bsa_lag-1_14                 ingress    lag-1                198.51.100.103           14               Any
120014
l2bsa_lag-1_15                 ingress    lag-1                198.51.100.103           15               Any
120015
l2bsa_lag-1_16                 ingress    lag-1                198.51.100.103           16               Any
120016
```

Example 8: Summary view of L2X information for a remote egress router

```
supervisor@rtbrick: op> show l2x  nexthop6 2001:db8:0:85::
Name                           Direction  Incoming Intf        Outgoing Intf/Next Hop    Outer VLAN
Inner VLAN      Service label
l2bsa_lag-1_11                 ingress    lag-1                2001:db8:0:85::          11               Any
120011
l2bsa_lag-1_12                 ingress    lag-1                2001:db8:0:85::          12               Any
120012
l2bsa_lag-1_13                 ingress    lag-1                2001:db8:0:85::          13               Any
120013
l2bsa_lag-1_14                 ingress    lag-1                2001:db8:0:85::          14               Any
120014
l2bsa_lag-1_15                 ingress    lag-1                2001:db8:0:85::          15               Any
120015
l2bsa_lag-1_16                 ingress    lag-1                2001:db8:0:85::          16               Any
120016
l2bsa_lag-1_17                 ingress    lag-1                2001:db8:0:85::          17               Any
120017
l2bsa_lag-1_18                 ingress    lag-1                2001:db8:0:85::          18               Any
120018
l2bsa_lag-1_19                 ingress    lag-1                2001:db8:0:85::          19               Any
120019
l2bsa_lag-1_20                 ingress    lag-1                2001:db8:0:85::          20               Any
120020
l2bsa_lag-1_21                 ingress    lag-1                2001:db8:0:85::          21               Any
120021
```

Example 9: L2X information in detailed format

```
supervisor@rtbrick: op> show l2x detail
L2X name: l2bsa_lag-1_11
  Direction: ingress
  Status: Download success
  Incoming interface: lag-1
  Service label: 120011
  Subtype: Incoming Port - Outer Vlan - Any Inner Vlan Match
  Incoming outer VLAN: 11
  Incoming inner VLAN: Any
  Ingress vlan operation:
    Vlan operation: Swap-Outer-Vlan
    Outer vlan: 64
  NextHop:
    NextHop IP: 2001:db8:0:85::
    Lookup instance: default
    Lookup AFI: ipv6
    Lookup SAFI: labeled-unicast
    NextHop type: Remote ingress cross connect
    NextHop action: mpls label push
  Egress vlan operation:
```

Example 10: Detailed L2X information for a specific L2X

```
supervisor@rtbrick: op> show l2x test1 detail
L2X name: test1
  Direction: ingress
  Status: Download success
  Incoming interface: ifp-0/0/4
  Outgoing interface: ifp-0/0/10
  Subtype: Incoming Port - Any Vlan Match
  Incoming outer VLAN: Any
  Incoming inner VLAN: Any
  Ingress vlan operation:
  NextHop:
    NextHop type: Local egress cross connect
    NextHop action: No vlan manipulation - l2 forward
  Egress vlan operation:
```

Example 11: Detailed L2X information for a specific type and L2X

```
supervisor@rtbrick: op> show l2x type local test1 detail
L2X name: test1
  Direction: ingress
  Status: Download success
  Incoming interface: ifp-0/0/4
  Outgoing interface: ifp-0/0/10
  Subtype: Incoming Port - Any Vlan Match
  Incoming outer VLAN: Any
  Incoming inner VLAN: Any
  Ingress vlan operation:
  NextHop:
    NextHop type: Local egress cross connect
    NextHop action: No vlan manipulation - l2 forward
  Egress vlan operation:
```

Example 12: Detailed L2X information for a specific direction and L2X

```
supervisor@rtbrick: op> show l2x direction egress test2 detail
L2X name: test2
  Direction: egress
  Status: Download success
  Outgoing interface: ifp-0/0/4
  Service label: 1234
  Subtype: Service Label Match
  Incoming outer VLAN: -
  Incoming inner VLAN: -
  Ingress vlan operation:
  NextHop:
    NextHop type: Remote egress cross connect
    NextHop action: No vlan manipulation - l2 forward
  Egress vlan operation:
```

Example 13: Statistics for all installed L2X

```
supervisor@rtbrick: op> show l2x statistics

L2X Name: l2x-test1/0
    Physical Interface Name: ifp-0/0/4
    Logical Interface Type: L2x ingress vlan interface
    Port-Mapping-Core: 0
    Vlan-Port-ID: 1149251592
    MPLS-Port-ID: N/A
    Counters:
        In-Forward-Packets: 57
        In-Forward-Bytes: 5700
        In-Drop-Packets: 0
        In-Drop-Bytes: 0
        Out-Forward-Packets: 0
        Out-Forward-Bytes: 0
        Out-Drop-Packets: 0
        Out-Drop-Bytes: 0
L2X Name: l2x-d3b529d74770f91fb2acf5e38da70eb9213473dd7e996c6a
    Physical Interface Name: ifp-0/0/10
    Logical Interface Type: L2x egress vlan interface
    Port-Mapping-Core: 0
    Vlan-Port-ID: 1149251591
    MPLS-Port-ID: N/A
    Counters:
        In-Forward-Packets: 0
        In-Forward-Bytes: 0
        In-Drop-Packets: 0
        In-Drop-Bytes: 0
        Out-Forward-Packets: 0
        Out-Forward-Bytes: 0
        Out-Drop-Packets: 0
        Out-Drop-Bytes: 0
```

To access the Operational State API that corresponds to this CLI, click here.

L2X Clear Commands

Clear commands allow to reset operational states.

L2X Statistics

This commands resets L2X statistics.

Syntax:

clear l2x statistics <l2x-name>

Attribute	Description
-	Without any option, the command clears all L2X statistics.
<l2x-name>	L2X name.

Example:

```
supervisor@rtbrick: op> clear l2x statistics l2x-test1/0
```

3.2. EVPN-VPWS

3.2.1. EVPN-VPWS Overview

Ethernet Virtual Private Network (EVPN) is a Layer 2 internetworking technology similar to BGP/MPLS IP VPN. EVPN uses extended BGP reachability information and advertisements between different Layer 2 networks at various sites in the control plane.

The EVPN Virtual Private Wire Service (VPWS) is a point-to-point (P2P) service that is built on the EVPN service architecture. EVPN-VPWS uses MPLS tunnels to traverse the backbone network. It offers a Layer 2 packet forwarding mode that connects access circuits (ACs) as per the specifications of RFC 8214.

Supported Standards

RFC Number	Description
7432	BGP MPLS-Based Ethernet VPN
8214	Virtual Private Wire Service Support in Ethernet VPN

EVPN VPWS Network Model

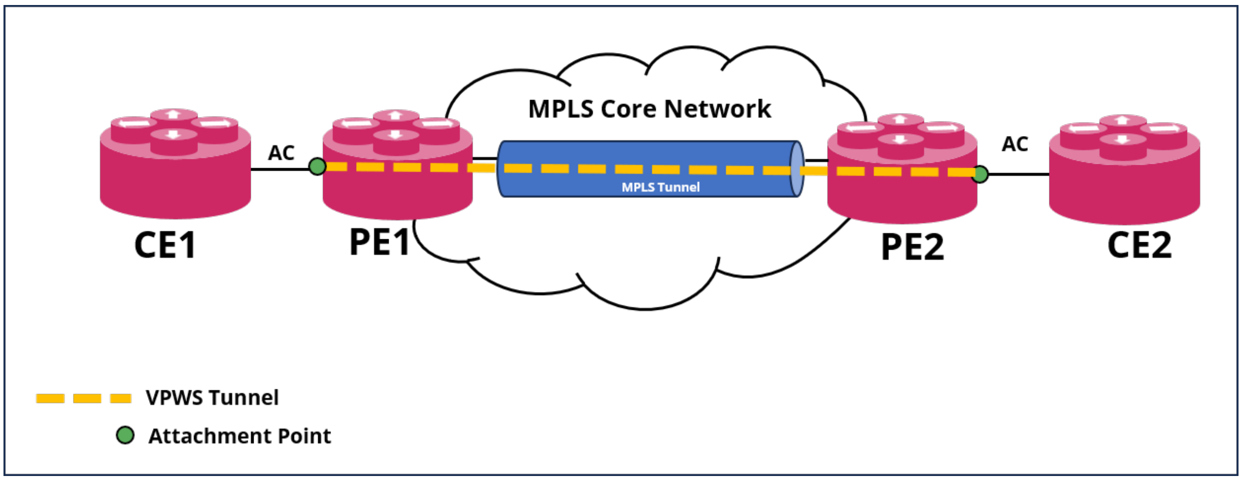
As shown in the figure below, an EVPN VPWS network contains the following building blocks:

- Customer edge (CE)—Customer device directly connected to the service provider network.
- Provider edge (PE)—Service provider device connected to CEs. PEs provide

access to the EVPN VPWS network and forward traffic between customer network sites by using public tunnels.

- Attachment circuit (AC)—A physical or virtual link between a CE and a PE.
- Pseudowire (PW)—A virtual bidirectional connection between two PEs. A PW comprises a pair of virtual links in opposite directions.
- MPLS transport tunnel—A connection that carries one or more PWs across the MPLS core or IP backbone, such as an MPLS tunnel.
- Cross-connect—A connection formed by two physical or virtual circuits, such as ACs and PWs, that switches packets between them.

The figure below shows the protocol packet exchange process in the EVPN VPWS.



PE1 and PE2 are each configured with an EVPN VPWS instance. After the PE receives packets from the AC, it adds the PW label and sends them to the peer PE through the MPLS label-switched path (LSP). After the other PE (PE2) receives the packet via the MPLS LSP, it pops the PW label of the packets and forwards the packets to the AC bound to the PW.

VLAN Tag Manipulation

VLAN Tag Manipulation involve modifying VLAN tags on Ethernet frames to ensure proper traffic segmentation and delivery across a Layer 2 network. RBFS supports some VLAN Tag Manipulation operations which are used to manipulate tags on Ethernet frames as they enter or leave a network interface. These manipulations include adding, removing, and rewriting VLAN tags. VLAN manipulation operations can be performed both at ingress and egress nodes.

The following VLAN manipulation operations are supported for Layer 2 VPN interfaces:

- **push**: Adds a single outer VLAN tag to an Ethernet frame.
- **pop**: Removes a single outer VLAN tag from an Ethernet tag frame.
- **swap**: Rewrite a single outer VLAN tag of an Ethernet tag frame.
- **push–push**: Adds two VLAN tags to an Ethernet frame.
- **pop-pop**: Removes two VLAN tags from an Ethernet tags frame.
- **swap–swap**: Rewrite both the outer and inner VLAN tags of an Ethernet tags frame.
- **swap–push**: Rewrite the outer VLAN tag and add a new inner VLAN tag for Ethernet tags' frame.
- **pop–swap**: Removes the outer VLAN tag and rewrites the inner VLAN tag of an Ethernet tags' frame.

Supported CLI Options for VLAN Tag Manipulation Operations

The following table presents the various VLAN tag manipulation operations and supported CLI options.

CLI Options	pop	push	swap	push-push	pop-pop	swap-push	swap-swap	pop-swap
vlan-id	No	Yes	Yes	Yes	No	Yes	Yes	Yes
vlan-encapsulation	No	Yes	Yes	Yes	No	Yes	Yes	Yes
innervlan-id	No	No	No	Yes	No	Yes	Yes	No

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Currently, the following features are not supported for EVPN VPWS:

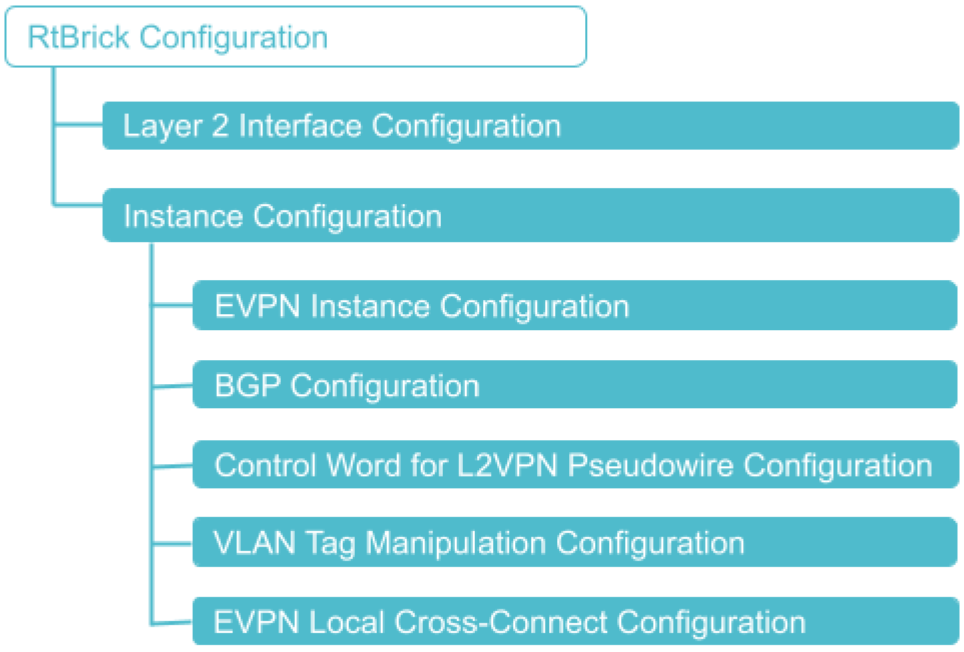
- Policy filtering options (at Address Family/Peer-Group/Instance levels)

- LAG interface as an attachment circuit
- LAG interface as an LSP MPLS path
- Route Reflection
- Add-Path
- VLAN change configuration on L2 IFL

3.2.2. EVPN VPWS Configuration

Configuration Hierarchy

The diagram illustrates the EVPN-VPWS configuration hierarchy. All EVPN VPWS configurations are done within an instance, such as the default instance or an EVPN service instance.


RtBrick Configuration
Layer 2 Interface Configuration
Instance Configuration
EVPN Instance Configuration
BGP Configuration
Control Word for L2VPN Pseudowire Configuration
VLAN Tag Manipulation Configuration
EVPN Local Cross-Connect Configuration

Configuration Syntax and Commands

The following sections describe the EVPN-VPWS configuration syntax and commands.

Layer 2 Interface Configuration

EVPN-VPWS supports the configuration of Layer 2 logical interfaces.

Syntax:

set interface <name> <attribute> <value>

Attribute	**Description**
<name>	Specify the name of the interface. Example: ifp-0/0/1.
unit <unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface
unit <unit-id> interface-type l2vpn-vpws	Specify the type of the L2VPN interface.
unit <unit-id> match-type port	Sets the "port" match-type for the L2VPN VPWS untagged interfaces. For tagged interfaces, "port" match-type is not supported. Ensure that there are no other tagged configurations for the interfaces when the port match-type is configured. When port matching is configured, it will match all traffic regardless of whether it is untagged, single-tagged, or double-tagged VLANs.
unit <unit-id> vlan <outer-vlan-id>	Outer VLAN ID.
unit <unit-id> vlan <inner-vlan-id>	Inner VLAN ID.
unit <unit-id> instance <instance>	Assign the logical interface to an instance.

The following example shows an untagged interface configuration along with "port" match-type for the L2VPN VPWS untagged interface.

```
set interface ifp-0/0/17 unit 0
set interface ifp-0/0/17 unit 0 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 0 match-type port
set interface ifp-0/0/17 unit 0 instance evpn-vpws-vrf1
commit
```

The following example shows a single-tagged interface configuration.

```
set interface ifp-0/0/17 unit 100
set interface ifp-0/0/17 unit 100 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 100 instance evpn-vpws-vrf2
set interface ifp-0/0/17 unit 100 vlan 100
```

```
commit
```

The following example shows a double-tagged interface configuration.

```
set interface ifp-0/0/17 unit 200
set interface ifp-0/0/17 unit 200 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 200 instance evpn-vpws-vrf3
set interface ifp-0/0/17 unit 200 vlan 200
set interface ifp-0/0/17 unit 200 inner-vlan 201
commit
```

EVPN Instance Configuration

An EVPN Instance (EVI) is a routing and forwarding instance of EVPN that covers all the participating PE routers in a VPN. EVI is configured per customer on the PE routers. Each EVI has a unique route distinguisher and one or more route targets.

Syntax:

set instance <name> <attribute> <value>

Attribute	Description
<name>	A unique name for the EVPN routing instance.
address-family <afi>	Address family identifier (AFI). Supported value: l2vpn
address-family <afi> <safi>	Subsequent address family identifier (SAFI). Supported values: evpn-vpws evpn-vpws needs to be enabled on the VRF instance.
protocol	Specifies the routing protocol
route-distinguisher <as-number | ipv4-address:id>	The route distinguisher (RD) uniquely defines routes within an IPv4 network. PE routers use route distinguishers to identify which VPN a packet belongs to. Supported formats are <as-number:id> or <ipv4-address:id>.
ipv4-router-id <ipv4-router-id>	The router ID of the routing instance.

Attribute	Description
route-target (import | export) <rt-value>	Route targets (RT) are used to transfer routes between VPN instances. The RT identifies a subset of routes that should be imported to or exported from a particular VPN instance. You can configure an RT for importing or exporting routes or both.

In the following configuration, VRF instance AFI has been set to AFI: l2vpn and SAFI: evpn-vpws.

```
set instance evpn-vpws-vrf1
set instance evpn-vpws-vrf1 ipv4-router-id 192.1.6.3
set instance evpn-vpws-vrf1 route-distinguisher 192.1.6.3:65006
set instance evpn-vpws-vrf1 address-family l2vpn evpn-vpws
set instance evpn-vpws-vrf1 address-family l2vpn evpn-vpws route-target import
target:192.1.6.0:65006
set instance evpn-vpws-vrf1 address-family l2vpn evpn-vpws route-target export
target:192.1.6.0:65006
commit
```

BGP Configuration

BGP L2VPN VFT (Virtual Forwarding Table) Configuration

Syntax:

set instance <name> **protocol bgp** <attribute> <value>

Attribute	Description
<name>	Name of the routing instance
address-family <afi>	Address family identifier (AFI). Supported value: l2vpn
address-family <afi> <safi>	Subsequent address family identifier (SAFI). Supported value: evpn-vpws ℹ evpn-vpws needs to be enabled on the VRF instance.
local-as <as-number>	The AS number in four-byte format. The numbers allowed are from 1 to 4294967295.
interface <name>	Interface that is bound to L2VPN

Attribute	Description
interface <name> local-service-id <local-service-id>	Specify the local service ID that is used to establish an EVPN PW between two PEs
interface <name> remote-service-id <remote-service-id>	Specify the remote service ID that is used to establish an EVPN PW between two PEs

The following example configures evpn-vpws-vrf1 instance for BGP L2VPN.

```
set instance evpn-vpws-vrf1 protocol bgp local-as 65006
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/0
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/0 local-service-id 100
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/0 remote-service-id 101
commit
```

BGP L2VPN EVPN Configuration

Configuring the BGP L2VPN EVPN Address Family

Syntax:

set instance <name> **protocol bgp address-family l2vpn evpn**

Attribute	Description
<name>	Name of the routing instance

To configure BGP L2VPN EVPN on the default instance, enter the following command:

```
set instance default protocol bgp address-family l2vpn evpn
commit
```

Configuring Address Families for Peer Groups

Syntax:

set instance <instance-name> **protocol bgp peer-group** <pg-name> **address-**

family <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported value: l2vpn.
<safi>	Subsequent address family identifier (SAFI). Supported value: evpn
extended-nexthop	Enable extended-next-hop encoding for BGP peer groups to allow the transfer of IPv4 prefixes over an IPv6 connection.
update-nexthop (ipv4-address | ipv6-address) <address>	BGP nexthop address for routes advertised to this peer group

To configure BGP L2VPN Peer Group on the default instance, enter the following command:

```
set instance default protocol bgp peer-group spine address-family l2vpn evpn
commit
```

Configuring Control Word for L2VPN Pseudowire

In a network with EVPN-VPWS service, a control word can be inserted between the label stack and the MPLS payload. Control word is disabled by default.

Syntax:

set instance <name> **protocol bgp address-family l2vpn evpn-vpws interface** <interface> **control-word enable**

Attribute	Description
<name>	Name of the routing instance.
<interface>	Name of the interface.

To enable Control Word on the default instance, enter the following command:

```
set instance default protocol bgp peer-group spine address-family l2vpn evpn-vpws
interface ifl-0/0/1/10 control-word enable
commit
```

Configuring VLAN Tag Manipulation

The following section describes the VLAN tag manipulation configuration syntax and commands.

Syntax:

set interface <interface-name> **unit** <unit id> **[input-vlan-map | output-vlan-map]** <attribute> <value>

Attribute	Description
<name>	Specify the name of the interface.
unit <unit-id>	Specify the unit ID for the interface.
unit <unit-id> input-vlan-map	Ingress VLAN mapping.
unit <unit-id> output-vlan-map	Egress VLAN mapping
vlan-operation	Specify any of the VLAN operations supported: pop, pop-pop, pop-swap, push, push-push, swap, swap-push, and swap-swap.
vlan-encapsulation	Specify the encapsulation type for the VLAN operation. Supported only 802.1q.
vlan-id	Specify the outer VLAN for the VLAN operation specified.
inner-vlan-id	Specify the inner VLAN for the VLAN operation specified.

Example for VLAN 'push' operation configuration

```
supervisor@rtbrick.net: cfg> show config set interface ifp-0/1/4 unit 1000
set interface ifp-0/1/4 unit 1000
set interface ifp-0/1/4 unit 1000 interface-type l2vpn-vpws
set interface ifp-0/1/4 unit 1000 output-vlan-map vlan-operation push
set interface ifp-0/1/4 unit 1000 output-vlan-map vlan-id 1001
set interface ifp-0/1/4 unit 1000 instance vrf2-evpn-vpws
set interface ifp-0/1/4 unit 1000 vlan 1000
```

Example: VLAN 'push' operation configuration.

```
supervisor@rtbrick.net: cfg> show config interface ifp-0/1/4 unit 1000
{
  "rtbrick-config:unit": [
    {
      "unit-id": 1000,
      "interface-type": "l2vpn-vpws",
      "output-vlan-map": {
        "vlan-operation": "push",
        "vlan-id": 1001
      },
      "instance": "vrf2-evpn-vpws",
      "vlan": 1000
    }
  ]
}
```

Configuring EVPN Local Cross-Connect

Local cross-connect is a special case of VPWS in which local and remote sites are connected to the same router. When the VPWS configuration is received, BGP checks if the exact remote site is locally configured. If so, BGP understands it needs to perform a local cross-connect. Local cross-connect for the local site and remote site and vice versa should be configured on the same VRF instance.

The following example shows a single-tagged interface configuration for the VRF instance (evpn-vpws-vrf1).

```
set interface ifp-0/0/17 unit 0
set interface ifp-0/0/17 unit 0 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 0 instance evpn-vpws-vrf1
set interface ifp-0/0/17 unit 0 vlan 100
set interface ifp-0/0/17 unit 10
set interface ifp-0/0/17 unit 10 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 10 instance evpn-vpws-vrf1
set interface ifp-0/0/17 unit 10 vlan 100
commit
```

The following examples show the local cross-connect configurations for the same VRF instance (evpn-vpws-vrf1).

```
set instance evpn-vpws-vrf1 protocol bgp local-as 65006
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/0
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/0 local-service-id 100
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/0 remote-service-id 101
commit
```

```
set instance evpn-vpws-vrf1 protocol bgp local-as 65006
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/10
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/10 local-service-id 101
set instance evpn-vpws-vrf1 protocol bgp address-family l2vpn evpn-vpws interface
ifl-0/0/17/10 remote-service-id 100
commit
```

3.2.3. EVPN VPWS Operational Commands

EVPN VPWS Show Commands

The following show commands display the EVPN VPWS-related information.

BGP Summary

This command displays BGP protocol parameters like attributes or timers that are generic to the BGP instance.

Syntax:

show bgp summary <option>

Attribute	Description
Option	Description
-	Without any option, the command displays the information for all instances.

Example: BGP summary for instance evpn-vpws-vrf1

```
supervisor@rtbrick: op> show bgp summary instance evpn-vpws-vrf1
Instance: evpn-vpws-vrf1
  General information
    Hostname: , Domain name:
    Local AS: 65006, Version: 4
    Local preference: 100, eBGP Protocol preference: 20, iBGP Protocol preference:
200
    Router ID: 192.168.6.10, Cluster ID: 192.168.6.10
  Capabilities
    Route refresh: True, AS4: True, Graceful restart: False, L2VPN EVPN-VPWS:True
  Best route selection
    Always compare MED: False, Ignore as path: False
    Ignore local preference: False, Ignore origin: False
    Ignore MED: False, Ignore route source: False
    Ignore router ID: False, Ignore uptime: True
```

```
      Ignore cluster length: False, Ignore peer IP: False
      Route select parameter: 0
    Timers
      Connect retry: 30s, Keepalive: 30s, Holdtime: 90s
    Statistics
      Peers configured: 0, Peers auto discovery: 0
        Peers in idle          : 0
        Peers in connect       : 0
        Peers in active        : 0
        Peers in opensent      : 0
        Peers in openconfirm   : 0
        Peers in established   : 0
```

BGP Peer

The 'show bgp peer' commands display information on BGP peers.

Syntax:

show bgp peer <option>

Option	Description
-	Without any option, the commands display all BGP peers in all instances in a summary table format.
detail	Detailed information on all BGP peers in all instances in a list view.
<peer-name>	Detailed information on the peer with the given name.
address <peer-address>	Detailed information on the peer with the given IP address.
history	Displays BGP peer history information such as the peer state down reasons.
history <peer-address>	Displays BGP peer history information such as the peer state down reasons for a specified peer.
instance <instance-name>	Summary of all BGP peers in the given instance.
instance <instance-name> detail	Detailed information on all BGP peers in the given instance.
instance <instance-name> detail <peer-name>	Detailed information on the peer with the given name in the given instance.

Option	Description
instance <instance-name> detail address <peer-address>	Detailed information on the peer with the given IP address in the given instance.
statistics	Received and sent BGP prefixes per AFI/SAFI for all peers in all instances.
statistics peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name.
statistics peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address.
statistics instance <instance-name> peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name in the given instance.
statistics instance <instance-name> peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address in the given instance.

Example 1: BGP Peer Information for instance default

```
supervisor@rtbrick: op> show bgp peer instance default
Instance: default
  Peer      Remote AS     State          Up/Down Time     PfxRcvd          PfxSent
  spine1    4200000000    Established    0d:00h:42m:53s   5                14
supervisor@rtbrick: op>
```

Example 2: Detailed view of BGP Peer Information

```
supervisor@rtbrick: op> show bgp peer detail
Peer: spine1, Peer IP: fe80::9a19:2cff:fe99:4701, Remote AS: 4200000000, Local:
fe80::9a19:2cff:fe36:3e03, Local AS: 4200000004, Any AS: False
  Type: ebgp, State: Established, Up/Down Time: 0d:01h:00m:40s
  Discovered on interface: ifl-0/0/2/1
  Last transition: Thu Apr 18 04:50:56 GMT +0000 2024, Flap count: 0
  Peer ID        : 192.1.0.1, Local ID  : 192.1.0.4
  Instance       : default, Peer group: spine
  6PE enabled    : False
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    Keepalive timer     : expires in 21s 796304us
    Holddown timer      : expires in 1m 13s 560627us
  NLRIs:
    Sent           : ['l2vpn-evpn', 'ipv4-unicast', 'ipv6-unicast', 'ipv4-vpn-
unicast', 'ipv6-vpn-unicast', 'ipv4-vpn-multicast', 'l2vpn-vpls', 'ipv4-labeled-
unicast', 'ipv6-labeled-unicast']
    Received       : ['l2vpn-evpn', 'ipv4-unicast', 'ipv6-unicast', 'ipv4-vpn-
unicast', 'ipv6-vpn-unicast', 'ipv4-vpn-multicast', 'l2vpn-vpls', 'ipv4-labeled-
unicast', 'ipv6-labeled-unicast']
    Negotiated     : ['l2vpn-evpn', 'ipv4-unicast', 'ipv6-unicast', 'ipv4-vpn-
unicast', 'ipv6-vpn-unicast', 'ipv4-vpn-multicast', 'l2vpn-vpls', 'ipv4-labeled-
unicast', 'ipv6-labeled-unicast']
  Capabilities:
    Addpath sent                  : None
    Addpath received              : None
    Addpath negotiated            : None
    Extended nexthop sent         : ['ipv4-unicast', 'ipv4-vpn-unicast', 'ipv4-
vpn-multicast', 'ipv4-labeled-unicast']
    Extended nexthop received     : ['ipv4-unicast', 'ipv4-labeled-unicast',
'ipv4-vpn-multicast', 'ipv4-vpn-unicast']
    Extended nexthop negotiated   : ['ipv4-unicast', 'ipv4-labeled-unicast',
'ipv4-vpn-multicast', 'ipv4-vpn-unicast']
    Capabilities:
      Feature                      Sent            Received        Negotiated
      Route refresh                True            True            True
      4 byte AS                    True            True            True
      Graceful restart             False           False           False
      Link local only              False           False           False
  Prefix Limit:
  End of RIB:
    Address family                 Sent                                Received
    IPv4 unicast                   Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv4 labeled-unicast           Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv6 unicast                   Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv6 labeled-unicast           Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv4 VPN-unicast               Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv6 VPN-unicast               Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv4 VPN-multicast             Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    L2VPN VPLS                     Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    L2VPN EVPN                     Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
  Message stats:
    Session stats:
      Direction   Open         Update       Keepalive    Notify       Route
refresh
      Input       0            0            72           0            0
      Output      0            0            71           0            0
    Total stats:
      Input       0            0            72           0            0
      Output      0            0            71           0            0
    Route stats:
      Address family               Received     Sent         Prefix limit Idle
timeout
      IPv4 unicast                 0            0            0            0
      IPv4 labeled-unicast         0            0            0            0
      IPv6 unicast                0            0            0            0
      IPv6 labeled-unicast        0            0            0            0
      IPv4 VPN-unicast            0            0            0            0
      IPv6 VPN-unicast            0            0            0            0
      IPv4 VPN-multicast          0            0            0            0
      L2VPN VPLS                  0            0            0            0
      L2VPN EVPN                  0            0            0            0
<...>
```

Example 3: BGP Peer Statistics for a specific peer

```
supervisor@rtbrick: op> show bgp peer statistics peer address
fe80::781a:c6ff:fec0:1
Instance: default
  Peer                            AFI     SAFI               PfxRcvd    PfxSent
  spine1                          ipv4    unicast            0          0
                                  ipv4    labeled-unicast    0          0
                                  ipv6    unicast            1          4
                                  ipv6    labeled-unicast    1          4
                                  ipv4    vpn-unicast        0          0
                                  ipv6    vpn-unicast        0          0
                                  ipv4    multicast          0          0
                                  ipv4    vpn-multicast      0          0
                                  l2vpn   evpn               3          6
```

Example 4: BGP Peer history for a specified peer

```
supervisor@rtbrick.net: op> show bgp peer history peer address 192:168::40
Instance: ip2vrf
  Peer Address                Source Address             Type             Last
Reset Reason
  192:168::40                 192:168:5::20              FSM Error        FSM
Error, Sub-Code: Unexpected message in OpenSent State
```

BGP RIB-in

This command displays the received routes.

Syntax:

show bgp rib-in <option>

Option	Description
-	Without any option, the command displays information on the received BGP routing table on all instances in a summary table format.

Option	Description
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4', 'ipv6' and l2vpn.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are evpn, 'labeled-unicast', 'unicast', 'vpn-multicast', 'vpn-unicast', and evpn-vpws.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> community <community-name>	BGP community details for the given AFI, SAFI, and instance.
<afi> <safi> error	BGP route with error status for the given AFI, SAFI, and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the BGP rib-in where you can find received information for the EVPN VPWS instances like evpn-vpws-vrf1, evpn-vpws-vrf2, and evpn-vpws-vrf3.

```
supervisor@rtbrick: op> show bgp rib-in
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv6, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 3
    Prefix                                  Next Hop                            MED        Lpref
AS Path    Status
    192:1::3/128                                                                0          100
-          Valid
```

```
    192:1::5/128                                                       0          100
-          Valid
    192:1::6/128                                                       0          100
-          Valid
  Hostname: spine1, Peer IP: fe80::781a:c6ff:fec0:1
  Source IP: fe80::7857:d6ff:fec0:0, Total routes: 1
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    192:1::1/128                           fe80::781a:c6ff:fec0:1      0          -
4200000000 Valid
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 3
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    192:1::3/128                                                       0          100
-          Valid
    192:1::5/128                                                       0          100
-          Valid
    192:1::6/128                                                       0          100
-          Valid
  Hostname: spine1, Peer IP: fe80::781a:c6ff:fec0:1
  Source IP: fe80::7857:d6ff:fec0:0, Total routes: 1
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    192:1::1/128                           fe80::781a:c6ff:fec0:1      0          -
4200000000 Valid
Instance: evpn-vpws-vrf1, AFI: l2vpn, SAFI: evpn-vpws
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 1
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:100/112                              0          100
-          Valid
Instance: evpn-vpws-vrf2, AFI: l2vpn, SAFI: evpn-vpws
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 1
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:110/112                              0          100
-          Valid
Instance: evpn-vpws-vrf3, AFI: l2vpn, SAFI: evpn-vpws
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 1
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:200/112                              0          100
-          Valid
Instance: default, AFI: l2vpn, SAFI: evpn
  Hostname: spine1, Peer IP: fe80::781a:c6ff:fec0:1
  Source IP: fe80::7857:d6ff:fec0:0, Total routes: 3
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:101/112  192:1::1                    0          -
4200000000 Valid
    00.00.00.00.00.00.00.00.00.00:111/112  192:1::1                    0          -
4200000000 Valid
    00.00.00.00.00.00.00.00.00.00:201/112  192:1::1                    0          -
4200000000 Valid
```

Example 2: Summary view of the BGP rib-in where you can find received information for address family l2vpn vpws.

```
supervisor@rtbrick: op> show bgp rib-in l2vpn evpn
Flags: & - Imported, ! - Error
Instance: default, AFI: l2vpn, SAFI: evpn
 Hostname: spine1, Peer IP: fe80::9a19:2cff:fe99:4701, Source IP: fe80::9a19:2cff:fe36:3e03, Total routes: 3
    Prefix                                 Next Hop                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:101/112  192:1::1                    0          -
4200000000, 4200000003 Valid
    00.00.00.00.00.00.00.00.00.00:111/112  192:1::1                    0          -
```

```
4200000000, 4200000003 Valid
    00.00.00.00.00.00.00.00.00.00:201/112        192:1::1                    0          -
4200000000, 4200000003 Valid
```

Example 3: Summary view of the BGP RIB in with the error flag.

```
supervisor@rtbrick.net: cfg> show bgp rib-in l2vpn vpls
Flags: & - Imported, ! - Error
Instance: default, AFI: l2vpn, SAFI: vpls
  Hostname: RR1, Peer IP: 192.168.0.30
  Source IP: 192.168.0.40, Total routes: 2
    Flags  Prefix                                Next Hop      MED  Lpref  AS
Path
           00.00.00.00.00.00.00.00.00.00:100/112  192.168.3.21  -    100
4200000001
           00.00.00.00.00.00.00.00.00.00:210/112  192.168.3.21  -    100
4200000001
```

BGP FIB

The 'show bgp fib' commands display the BGP forwarding table.

Syntax:

show bgp fib <option>

Option	Description
-	Without any option, the commands display the BGP forwarding table for all address families and all instances in a summary table format.
<afi>	BGP forwarding table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are l2vpn, 'ipv4' and 'ipv6'.
<afi> <safi>	BGP forwarding table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'unicast', 'labeled-unicast', 'vpn-multicast', 'vpn-unicast', and evpn-vpws.
<afi> <safi> detail	Detailed list view of the BGP forwarding table for the given address family (AFI) and sub-address family (SAFI), and all instances.

Option	Description
<afi> <safi> <prefix>	BGP forwarding table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP forwarding table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP forwarding table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP forwarding table entry for the given prefix and instance.

```
supervisor@rtbrick: op> show bgp fib
Instance: default, AFI: ipv6, SAFI: unicast
  Prefix                                        Preference      Out Label          Next Hop
  192:1::1/128                                  20              -                  fe80::781a:c6ff:fec0:1
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Prefix                                        Preference      Out Label          Next Hop
  192:1::1/128                                  20              -                  fe80::781a:c6ff:fec0:1
Instance: evpn-vpws-vrf1, AFI: l2vpn, SAFI: evpn-vpws
  Prefix                                        Preference      Out Label          Next Hop
  00.00.00.00.00.00.00.00.00.00:101/112         20              20004,bos:1        192:1::1
Instance: evpn-vpws-vrf2, AFI: l2vpn, SAFI: evpn-vpws
  Prefix                                        Preference      Out Label          Next Hop
  00.00.00.00.00.00.00.00.00.00:111/112         20              20005,bos:1        192:1::1
Instance: evpn-vpws-vrf3, AFI: l2vpn, SAFI: evpn-vpws
  Prefix                                        Preference      Out Label          Next Hop
  00.00.00.00.00.00.00.00.00.00:201/112         20              20006,bos:1        192:1::1
```

BGP RIB-out

This command displays the routes that were advertised to peers.

Syntax:

show bgp rib-out <option>

Option	Description
-	Without any option, the command displays advertised BGP routes for all instances.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are l2vpn, 'ipv4' and 'ipv6'.

Option	Description
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are evpn, 'unicast', 'labeled-unicast', 'multicast', and 'vpn-unicast'.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the routes advertised to a peer

```
supervisor@rtbrick: op> show bgp rib-out
Instance: default, AFI: ipv6, SAFI: unicast
  Peer-group: spine, Sent routes: 4
    Prefix                                  MED             Lpref             Origin          Next Hop
AS Path
    192:1::1/128                            0               -                 Incomplete      -
4200000003, 4200000000
    192:1::3/128                            0               -                 Incomplete      -
4200000003
    192:1::5/128                            0               -                 Incomplete      -
4200000003
    192:1::6/128                            0               -                 Incomplete      -
4200000003
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Peer-group: spine, Sent routes: 4
    Prefix                                  MED             Lpref             Origin          Next Hop
AS Path
    192:1::1/128                            0               -                 Incomplete      -
4200000003, 4200000000
    192:1::3/128                            0               -                 Incomplete      -
4200000003
    192:1::5/128                            0               -                 Incomplete      -
4200000003
    192:1::6/128                            0               -                 Incomplete      -
4200000003
Instance: default, AFI: l2vpn, SAFI: evpn
  Peer-group: spine, Sent routes: 6
    Prefix                                  MED             Lpref             Origin          Next Hop
AS Path
    00.00.00.00.00.00.00.00.00.00:100/112   0               -                 Incomplete      192:1::3
```

```
4200000003
    00.00.00.00.00.00.00.00.00.00:101/112         0               -                 Incomplete      192:1::3
4200000003, 4200000000
    00.00.00.00.00.00.00.00.00.00:110/112         0               -                 Incomplete      192:1::3
4200000003
    00.00.00.00.00.00.00.00.00.00:111/112         0               -                 Incomplete      192:1::3
4200000003, 4200000000
    00.00.00.00.00.00.00.00.00.00:200/112         0               -                 Incomplete      192:1::3
4200000003
    00.00.00.00.00.00.00.00.00.00:201/112         0               -                 Incomplete      192:1::3
4200000003, 4200000000
```

BGP L2VPN

This command displays the BGP L2VPN information.

Syntax:

show bgp l2vpn <option>

Option	Description
pseudowire	Displays pseudowire information for all instances.
pseudowire instance <name>	Displays pseudowire information for the specified instance.

Example: Display L2VPN pseudowire information for instance evpn-vpws-vrf1

```
supervisor@rtbrick: op> show bgp l2vpn pseudowire instance evpn-vpws-vrf1
   Instance: evpn-vpws-vrf1 AFI: l2vpn, SAFI: evpn-vpws
   Peer IP: 12.0.0.2
   Route Distinguisher: 192.168.6.10:65006
   Number of local interfaces: 1
    Interface name  Prefix                                Status         Local SID   Remote SID   Pop
Label            Push Label
    ifl-0/0/3/0  00.00.00.00.00.00.00.00.00.00:100/112      up             100         101
label:20004,bos:1    label:20004,bos:1
```

3.3. BGP-signaled L2VPN

3.3.1. BGP-signaled L2VPN Overview

The BGP-signaled L2VPN uses BGP for signaling and auto-discovery to establish multipoint Layer 2 VPN over the MPLS backbone network. The remote cross-connect is a point-to-point (P2P) service that connects two locations using the MPLS core network and MP-BGP. The remote cross-connect uses MPLS tunnels to traverse the backbone network. It offers a Layer 2 packet forwarding mode that connects access circuits (ACs). The RBFS implementation of BGP-signaled L2VPN is in accordance with RFC-6624 which supports L2VPN using BGP for auto-discovery and Signaling.

Supported Standards

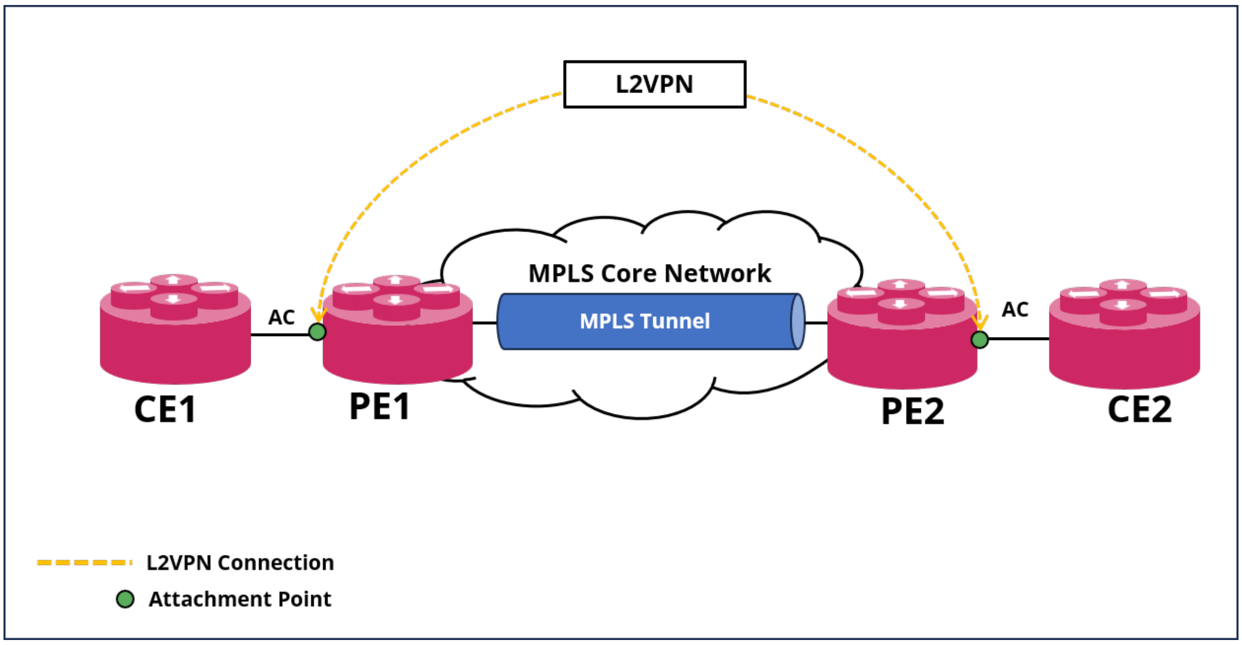
RFC Number	Description
6624	Layer 2 Virtual Private Networks Using BGP for Auto-Discovery and Signaling

BGP-signaled L2VPN Network Model

As shown in the figure below, a BGP-signaled L2VPN network contains the following building blocks:

- Customer edge (CE)—Customer device directly connected to the service provider network.
- Provider edge (PE)—Service provider device connected to CEs. PEs provide access to the Pseudowire network and forward traffic between customer network sites by using public tunnels.
- Attachment circuit (AC)—A physical or virtual link between a CE and a PE.
- Pseudowire (PW)—A virtual bidirectional connection between two PEs. A PW comprises a pair of virtual links in opposite directions.
- MPLS transport tunnel—A connection that carries one or more PWs across the MPLS core or IP backbone, such as an MPLS tunnel.
- Cross-connect—A connection formed by two physical or virtual circuits, such as ACs and PWs, that switches packets between them.

The figure below shows the protocol packet exchange process in the BGP-signaled L2VPN.



PE1 and PE2 are each configured with a BGP L2VPN instance. After the PE receives packets from the AC, it adds the PW label and sends them to the peer PE through the MPLS transport tunnel. After the other PE receives the packet via the MPLS transport path, it removes the PW label of the packets and forwards the packets to the AC bound to the PW.

VLAN Tag Manipulation

VLAN Tag Manipulation involve modifying VLAN tags on Ethernet frames to ensure proper traffic segmentation and delivery across a Layer 2 network. RBFS supports some VLAN Tag Manipulation operations which are used to manipulate tags on Ethernet frames as they enter or leave a network interface. These manipulations include adding, removing, and rewriting VLAN tags. VLAN manipulation operations can be performed both at ingress and egress nodes.

The following VLAN manipulation operations are supported for Layer 2 VPN interfaces:

- **push**: Adds a single outer VLAN tag to an Ethernet frame.
- **pop**: Removes a single outer VLAN tag from an Ethernet tag frame.
- **swap**: Rewrite a single outer VLAN tag of an Ethernet tag frame.
- **push–push**: Adds two VLAN tags to an Ethernet frame.
- **pop-pop**: Removes two VLAN tags from an Ethernet tags frame.
- **swap–swap**: Rewrite both the outer and inner VLAN tags of an Ethernet tags

frame.

- **swap–push**: Rewrite the outer VLAN tag and add a new inner VLAN tag for Ethernet tags' frame.
- **pop–swap**: Removes the outer VLAN tag and rewrites the inner VLAN tag of an Ethernet tags' frame.

Supported CLI Options for VLAN Tag Manipulation Operations

The following table presents the various VLAN tag manipulation operations and supported CLI options.

CLI Options	pop	push	swap	push-push	pop-pop	swap-push	swap-swap	pop-swap
vlan-id	No	Yes	Yes	Yes	No	Yes	Yes	Yes
vlan-encapsulation	No	Yes	Yes	Yes	No	Yes	Yes	Yes
innervlan-id	No	No	No	Yes	No	Yes	Yes	No

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Unsupported Features

The following features are not supported for the BGP-signaled L2VPN:

- Policy filtering options (at Address Family/Peer-Group/Instance levels)
- LAG interface as an attachment circuit
- LAG interface as an LSP MPLS path
- Route Reflection
- Add-Path
- VLAN change configuration on L2 IFL

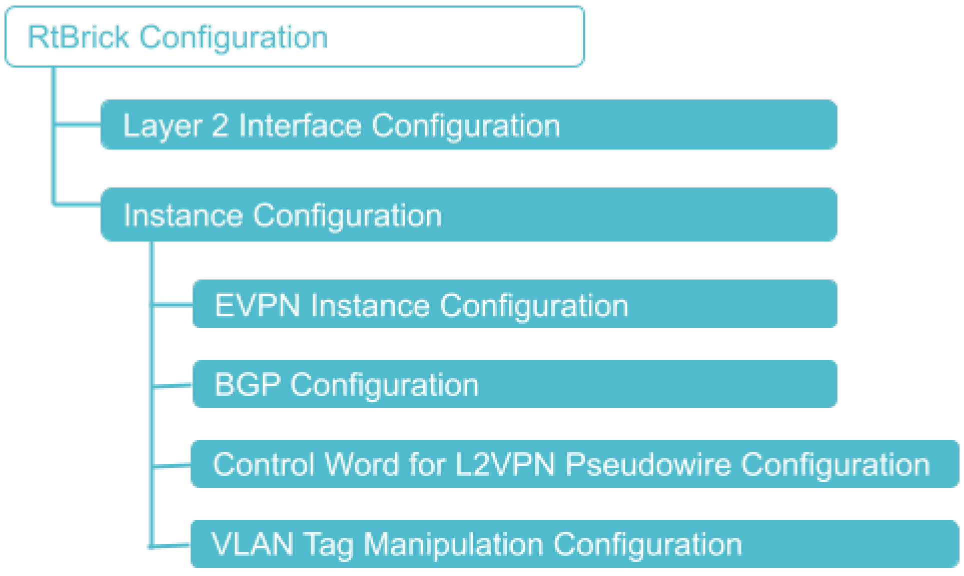
Guidelines and Limitations

- The control-word flag is disabled by default in Layer 2 Attributes Extended Community. To ensure interoperability with RBFS, the control-word flag needs to be disabled.

3.3.2. Configuring BGP-signaled L2VPN

Configuration Hierarchy

The diagram illustrates the BGP-signaled L2VPN configuration hierarchy. All BGP-signaled L2VPN configurations are done within an instance, such as the default instance or an EVPN service instance.


RtBrick Configuration
Layer 2 Interface Configuration
Instance Configuration
EVPN Instance Configuration
BGP Configuration
Control Word for L2VPN Pseudowire Configuration
VLAN Tag Manipulation Configuration

Configuration Syntax and Commands

The following sections describe the BGP-signaled L2VPN configuration syntax and commands.

Layer 2 Interface Configuration

BGP-signaled L2VPN supports the configuration of Layer 2 logical interfaces.

Syntax:

set interface <name> <attribute> <value>

Attribute	Description
<name>	Specify the name of the interface. Example: ifp-0/0/1.
unit <unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface
unit <unit-id> interface-type l2vpn-vpws	Specify the type of the L2VPN interface.
unit <unit-id> match-type port	Sets the "port" match-type for the L2VPN VPWS untagged interfaces. For tagged interfaces, "port" match-type is not supported. Ensure that there are no other tagged configurations for the interfaces when the port match-type is configured. When port matching is configured, it will match all traffic regardless of whether it is untagged, single-tagged, or double-tagged VLANs.
unit <unit-id> vlan <outer-vlan-id>	Outer VLAN ID.
unit <unit-id> vlan <inner-vlan-id>	Inner VLAN ID.
unit <unit-id> instance <instance>	Assign the logical interface to an instance.

The following example shows an untagged interface configuration along with "port" match-type for the L2VPN VPWS untagged interface.

```
set interface ifp-0/0/17 unit 0
set interface ifp-0/0/17 unit 0 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 0 match-type port
set interface ifp-0/0/17 unit 0 instance evpn-vpws-vrf1
commit
```

The following example shows a single-tagged interface configuration:

```
set interface ifp-0/0/17 unit 100
set interface ifp-0/0/17 unit 100 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 100 instance l2vpn_ce1
set interface ifp-0/0/17 unit 100 vlan 100
commit
```

The following example shows a double-tagged interface configuration:

```
set interface ifp-0/0/17 unit 200
set interface ifp-0/0/17 unit 200 interface-type l2vpn-vpws
set interface ifp-0/0/17 unit 200 instance l2vpn_ce2
set interface ifp-0/0/17 unit 200 vlan 200
set interface ifp-0/0/17 unit 200 inner-vlan 201
commit
```

EVPN Instance Configuration

An EVPN Instance (EVI) is a routing and forwarding instance of EVPN that covers all the participating PE routers in a VPN. EVI is configured per customer on the PE routers. Each EVI has a unique route distinguisher and one or more route targets.

Syntax:

set instance <name> <attribute> <value>

Attribute	Description
<name>	A unique name for the EVPN routing instance.
address-family <afi>	Address family identifier (AFI). Supported value: l2vpn
address-family <afi> <safi>	Subsequent address family identifier (SAFI). Supported values: vpls-vpws The SAFI vpls-vpws needs to be enabled on the VRF instance.
protocol	Specifies the routing protocol
route-distinguisher <as-number | ipv4-address:id>	The route distinguisher (RD) uniquely defines routes within an IPv4 network. PE routers use route distinguishers to identify which VPN a packet belongs to. Supported formats are <as-number:id> or <ipv4-address:id>.
ipv4-router-id <ipv4-router-id>	The router ID of the routing instance.
route-target (import | export) <rt-value>	Route targets (RT) are used to transfer routes between VPN instances. The RT identifies a subset of routes that should be imported to or exported from a particular VPN instance. You can configure an RT for importing or exporting routes or both.

In the following configuration, VRF instance AFI has been set to AFI: l2vpn and SAFI: vpls-vpws.

```
set instance l2vpn_ce1
set instance l2vpn_ce1 ipv4-router-id 182.1.6.4
set instance l2vpn_ce1 route-distinguisher 182.1.6.4:65006
set instance l2vpn_ce1 address-family l2vpn vpls-vpws
set instance l2vpn_ce1 address-family l2vpn vpls-vpws route-target import
target:182.1.6.0:65006
set instance l2vpn_ce1 address-family l2vpn vpls-vpws route-target export
target:182.1.6.0:65006
commit
```

BGP Configuration

BGP L2VPN VFT (Virtual Forwarding Table) Configuration

Syntax:

set instance <name> **protocol bgp** <attribute> <value>

Attribute	**Description**
<name>	Name of the routing instance
address-family <afi>	Address family identifier (AFI). Supported value: l2vpn
address-family <afi> <safi>	Subsequent address family identifier (SAFI). Supported value: vpls-vpws ℹ The SAFI vpls-vpws needs to be enabled on the VFT instance.
local-as <as-number>	The AS number in four-byte format. The numbers allowed are from 1 to 4294967295.
interface <name>	Interface that is bound to L2VPN
interface <name> local-site-id <local-site-id>	Specify the local site ID that is used to establish an EVPN PW between two PEs
interface <name> remote-site-id <remote-site-id>	Specify the remote site ID that is used to establish an EVPN PW between two PEs
interface <name> label-base <value>	Specify the label base value.

The following example configures l2vpn_ce1 instance for BGP L2VPN.

```
set instance l2vpn_ce1 protocol bgp local-as 65006
set instance l2vpn_ce1 protocol bgp address-family l2vpn vpls-vpws
set instance l2vpn_ce1 protocol bgp address-family l2vpn vpls-vpws interface ifl-
0/0/17/100
set instance l2vpn_ce1 protocol bgp address-family l2vpn vpls-vpws interface ifl-
0/0/17/100 local-site-id 300
set instance l2vpn_ce1 protocol bgp address-family l2vpn vpls-vpws interface ifl-
0/0/17/100 remote-site-id 301
set instance l2vpn_ce1 protocol bgp address-family l2vpn vpls-vpws interface ifl-
0/0/17/100 label-base 150000
commit
```

BGP L2VPN Configuration

Configuring the BGP L2VPN Address Family

Syntax:

set instance <name> **protocol bgp address-family l2vpn vpls**

Attribute	Description
<name>	Name of the routing instance

To configure BGP L2VPN VPLS on the default instance, enter the following command:

```
set instance default protocol bgp address-family l2vpn vpls
commit
```

Configuring Address Families for Peer Groups

Syntax:

set instance <instance-name> **protocol bgp peer-group** <pg-name> **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported value: l2vpn.
<safi>	Subsequent address family identifier (SAFI). Supported value: vpls

Attribute	Description
extended-nexthop	Enable extended-next-hop encoding for BGP peer groups to allow the transfer of IPv4 prefixes over an IPv6 connection.
update-nexthop (ipv4-address | ipv6-address) <address>	BGP nexthop address for routes advertised to this peer group

To configure BGP L2VPN Peer Group on the default instance, enter the following command:

```
set instance default protocol bgp peer-group spine address-family l2vpn vpls
commit
```

Configuring Control Word for L2VPN Pseudowire

A control word can be inserted between the label stack and the MPLS payload. Control word is disabled by default.

Syntax:

set instance <name> **protocol bgp address-family l2vpn vpls-vpws interface** <interface> **control-word enable**

Attribute	Description
<name>	Name of the routing instance.
<interface>	Name of the interface.

Example: Enter the following command to enable Control Word on the default instance.

```
set instance default protocol bgp peer-group spine address-family l2vpn vpls-vpws
interface ifl-0/0/1/10 control-word enable
commit
```

Configuring VLAN Tag Manipulation

The following section describes the VLAN tag manipulation configuration syntax and commands.

Syntax:

set interface <interface-name> **unit** <unit id> **[input-vlan-map | output-vlan-map]** <attribute> <value>

Attribute	Description
<name>	Specify the name of the interface.
unit <unit-id>	Specify the unit ID for the interface.
unit <unit-id> input-vlan-map	Ingress VLAN mapping.
unit <unit-id> output-vlan-map	Egress VLAN mapping
vlan-operation	Specify any of the VLAN operations supported: pop, pop-pop, pop-swap, push, push-push, swap, swap-push, and swap-swap.
vlan-encapsulation	Specify the encapsulation type for the VLAN operation. Supported only 802.1q.
vlan-id	Specify the outer VLAN for the VLAN operation specified.
inner-vlan-id	Specify the inner VLAN for the VLAN operation specified.

Example for VLAN operation 'push' configuration

```
supervisor@rtbrick.net: cfg> show config set interface ifp-0/1/4 unit 1000
set interface ifp-0/1/4 unit 1000
set interface ifp-0/1/4 unit 1000 interface-type l2vpn-vpws
set interface ifp-0/1/4 unit 1000 input-vlan-map vlan-operation push
set interface ifp-0/1/4 unit 1000 input-vlan-map vlan-id 1001
set interface ifp-0/1/4 unit 1000 instance l2vpn_ce1
set interface ifp-0/1/4 unit 1000 vlan 1000
```

Example: VLAN operation 'push' configuration output

```
supervisor@rtbrick.net: cfg> show config interface ifp-0/1/4 unit 1000
{
  "rtbrick-config:unit": [
    {
      "unit-id": 1000,
      "interface-type": "l2vpn-vpws",
      "input-vlan-map": {
        "vlan-operation": "push",
        "vlan-id": 1001
      },
      "instance": "l2vpn_ce1",
      "vlan": 1000
    }
  ]
}
```

Example for VLAN operation 'push-push' configuration

```
supervisor@rtbrick.net: cfg> show config set interface ifp-0/1/5 unit 2000
set interface ifp-0/1/5 unit 2000
set interface ifp-0/1/5 unit 2000 interface-type l2vpn-vpws
set interface ifp-0/1/5 unit 2000 input-vlan-map vlan-operation push-push
set interface ifp-0/1/5 unit 2000 input-vlan-map vlan-id 2000
set interface ifp-0/1/5 unit 2000 input-vlan-map inner-vlan-id 2001
set interface ifp-0/1/5 unit 2000 instance l2vpn_ce1
set interface ifp-0/1/5 unit 2000 vlan 2000
set interface ifp-0/1/5 unit 2000 inner-vlan 2
```

Example: VLAN operation 'push-push' configuration

```
supervisor@rtbrick.net: cfg> show config interface ifp-0/1/5 unit 2000
{
  "rtbrick-config:unit": [
    {
      "unit-id": 2000,
      "interface-type": "l2vpn-vpws",
      "input-vlan-map": {
        "vlan-operation": "push-push",
        "vlan-id": 2000,
        "inner-vlan-id": 2001
      },
      "instance": "l2vpn_ce1",
      "vlan": 2000,
      "inner-vlan": 2
    }
  ]
}
```

3.3.3. BGP L2VPN Operational Commands

BGP L2VPN Show Commands

The following show commands display the BGP L2VPN-related information.

BGP Summary

This command displays BGP protocol parameters like attributes or timers that are generic to the BGP instance.

Syntax:

show bgp summary <option>

Attribute	**Description**
Option	Description
-	Without any option, the command displays the information for all instances.

Example: BGP summary for instance l2vpn_ce1

```
supervisor@rtbrick: op> show bgp summary instance l2vpn_ce1
Instance: l2vpn_ce1
  General information
    Hostname: , Domain name:
    Local AS: 65006, Version: 4
    Local preference: 100, eBGP Protocol preference: 20, iBGP Protocol preference:
200
    Router ID: 182.1.6.4, Cluster ID: 182.1.6.4
  Capabilities
    Route refresh: True, AS4: True, Graceful restart: False, L2VPN VPLS-VPWS:True
  Best route selection
    Always compare MED: False, Ignore as path: False
    Ignore local preference: False, Ignore origin: False
    Ignore MED: False, Ignore route source: False
    Ignore router ID: False, Ignore uptime: True
    Ignore cluster length: False, Ignore peer IP: False
    Route select parameter: 0
  Timers
    Connect retry: 30s, Keepalive: 30s, Holdtime: 90s
  Statistics
    Peers configured: 0, Peers auto discovery: 0
      Peers in idle          : 0
      Peers in connect       : 0
      Peers in active        : 0
      Peers in opensent      : 0
      Peers in openconfirm   : 0
```

BGP Peer

The 'show bgp peer' commands display information on BGP peers.

Syntax:

show bgp peer <option>

Option	Description
-	Without any option, the commands display all BGP peers in all instances in a summary table format.
detail	Detailed information on all BGP peers in all instances in a list view.
<peer-name>	Detailed information on the peer with the given name.
address <peer-address>	Detailed information on the peer with the given IP address.
history	Displays BGP peer history information such as the peer state down reasons.
history <peer-address>	Displays BGP peer history information such as the peer state down reasons for a specified peer.
instance <instance-name>	Summary of all BGP peers in the given instance.
instance <instance-name> detail	Detailed information on all BGP peers in the given instance.
instance <instance-name> detail <peer-name>	Detailed information on the peer with the given name in the given instance.
instance <instance-name> detail address <peer-address>	Detailed information on the peer with the given IP address in the given instance.
statistics	Received and sent BGP prefixes per AFI/SAFI for all peers in all instances.
statistics peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name.
statistics peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address.
statistics instance <instance-name> peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name in the given instance.
statistics instance <instance-name> peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address in the given instance.

Example 1: BGP Peer Information for instance default

```
supervisor@rtbrick: op> show bgp peer instance default
Instance: default
  Peer         Remote AS    State         Up/Down Time       PfxRcvd     PfxSent
  spine1       4200000000   Established   0d:00h:59m:45s     55          77
  spine1       4200000000   Established   0d:00h:59m:48s     55          77
supervisor@rtbrick: op>
```

Example 2: Detailed view of BGP Peer Information

```
supervisor@rtbrick: op> show bgp peer detail
Peer: spine1, Peer IP: fe80::9a19:2cff:fe99:4701, Remote AS: 4200000000, Local:
fe80::9a19:2cff:fe36:3e03, Local AS: 4200000004, Any AS: False
  Type: ebgp, State: Established, Up/Down Time: 0d:01h:00m:40s
  Discovered on interface: ifl-0/0/2/1
  Last transition: Thu Apr 18 04:50:56 GMT +0000 2024, Flap count: 0
  Peer ID        : 192.1.0.1, Local ID  : 192.1.0.4
  Instance       : default, Peer group: spine
  6PE enabled    : False
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    Keepalive timer     : expires in 21s 796304us
    Holddown timer      : expires in 1m 13s 560627us
  NLRIs:
    Sent           : ['l2vpn-evpn', 'ipv4-unicast', 'ipv6-unicast', 'ipv4-vpn-
unicast', 'ipv6-vpn-unicast', 'ipv4-vpn-multicast', 'l2vpn-vpls', 'ipv4-labeled-
unicast', 'ipv6-labeled-unicast']
    Received       : ['l2vpn-evpn', 'ipv4-unicast', 'ipv6-unicast', 'ipv4-vpn-
unicast', 'ipv6-vpn-unicast', 'ipv4-vpn-multicast', 'l2vpn-vpls', 'ipv4-labeled-
unicast', 'ipv6-labeled-unicast']
    Negotiated     : ['l2vpn-evpn', 'ipv4-unicast', 'ipv6-unicast', 'ipv4-vpn-
unicast', 'ipv6-vpn-unicast', 'ipv4-vpn-multicast', 'l2vpn-vpls', 'ipv4-labeled-
unicast', 'ipv6-labeled-unicast']
  Capabilities:
    Addpath sent                 : None
    Addpath received             : None
    Addpath negotiated           : None
    Extended nexthop sent        : ['ipv4-unicast', 'ipv4-vpn-unicast', 'ipv4-
vpn-multicast', 'ipv4-labeled-unicast']
    Extended nexthop received    : ['ipv4-unicast', 'ipv4-labeled-unicast',
'ipv4-vpn-multicast', 'ipv4-vpn-unicast']
    Extended nexthop negotiated  : ['ipv4-unicast', 'ipv4-labeled-unicast',
'ipv4-vpn-multicast', 'ipv4-vpn-unicast']
    Capabilities:
      Feature                     Sent            Received        Negotiated
      Route refresh               True            True            True
      4 byte AS                   True            True            True
      Graceful restart            False           False           False
      Link local only             False           False           False
  Prefix Limit:
  End of RIB:
    Address family                Sent                                Received
    IPv4 unicast                  Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv4 labeled-unicast          Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
```

```
04:50:59 GMT +0000 2024
    IPv6 unicast                   Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv6 labeled-unicast           Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv4 VPN-unicast               Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv6 VPN-unicast               Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    IPv4 VPN-multicast             Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    L2VPN VPLS                     Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
    L2VPN EVPN                     Thu Apr 18 04:50:59 GMT +0000 2024  Thu Apr 18
04:50:59 GMT +0000 2024
  Message stats:
    Session stats:
      Direction   Open         Update       Keepalive    Notify       Route
refresh
      Input       0            0            72           0            0
      Output      0            0            71           0            0
    Total stats:
      Input       0            0            72           0            0
      Output      0            0            71           0            0
    Route stats:
      Address family                Received     Sent         Prefix limit Idle
timeout
      IPv4 unicast                  0            0            0            0
      IPv4 labeled-unicast          0            0            0            0
      IPv6 unicast                  0            0            0            0
      IPv6 labeled-unicast          0            0            0            0
      IPv4 VPN-unicast              0            0            0            0
      IPv6 VPN-unicast              0            0            0            0
      IPv4 VPN-multicast            0            0            0            0
      L2VPN VPLS                    0            0            0            0
      L2VPN EVPN                    0            0            0            0
<...>
```

Example 3: BGP Peer Statistics for a specific peer

```
supervisor@rtbrick: op> show bgp peer statistics peer address
fe80::781a:c6ff:fec0:1
Instance: default
  Peer                            AFI     SAFI                PfxRcvd     PfxSent
  spine1                          ipv4    unicast             4           5
                                  ipv4    labeled-unicast     4           5
                                  ipv6    unicast             4           5
                                  ipv6    labeled-unicast     4           5
                                  ipv4    vpn-unicast         14          18
                                  ipv6    vpn-unicast         14          18
                                  ipv4    multicast           0           0
                                  ipv4    vpn-multicast       4           7
                                  l2vpn   vpls                3           6
                                  l2vpn   evpn                4           8
```

Example 4: BGP Peer history for a specified peer

```
supervisor@rtbrick.net: op> show bgp peer history peer address 192:168::40
Instance: ip2vrf
  Peer Address                  Source Address                Type             Last
Reset Reason
  192:168::40                   192:168:5::20                 FSM Error        FSM
Error, Sub-Code: Unexpected message in OpenSent State
```

BGP RIB-in

This command displays the received routes.

Syntax:

show bgp rib-in <option>

Option	Description
-	Without any option, the command displays information on the received BGP routing table on all instances in a summary table format.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4', 'ipv6' and l2vpn.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are evpn, 'labeled-unicast', 'unicast', 'vpn-multicast', 'vpn-unicast', and vpls.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.

Option	Description
<afi> <safi> community <community-name>	BGP community details for the given AFI, SAFI, and instance.
<afi> <safi> error	BGP route with error status for the given AFI, SAFI, and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the BGP rib-in where you can find received information for the BGP L2VPN instances like l2vpn_ce1, l2vpn_ce2, and l2vpn_ce3.

```
supervisor@rtbrick: op> show bgp rib-in
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv6, SAFI: unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 3
    Prefix                                       Next Hop                             MED         Lpref
AS Path    Status
    192:1::3/128                                                                      0           100
-          Valid
    192:1::5/128                                                                      0           100
-          Valid
    192:1::6/128                                                                      0           100
-          Valid
  Hostname: spine1, Peer IP: fe80::781a:c6ff:fec0:1
  Source IP: fe80::7857:d6ff:fec0:0, Total routes: 1
    Prefix                                       Next Hop                             MED         Lpref
AS Path    Status
    192:1::1/128                                 fe80::781a:c6ff:fec0:1               0           -
4200000000 Valid
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 3
    Prefix                                       Next Hop                             MED         Lpref
AS Path    Status
    192:1::3/128                                                                      0           100
-          Valid
    192:1::5/128                                                                      0           100
-          Valid
    192:1::6/128                                                                      0           100
-          Valid
  Hostname: spine1, Peer IP: fe80::781a:c6ff:fec0:1
  Source IP: fe80::7857:d6ff:fec0:0, Total routes: 1
    Prefix                                       Next Hop                             MED         Lpref
AS Path    Status
    192:1::1/128                                 fe80::781a:c6ff:fec0:1               0           -
4200000000 Valid
Instance: l2vpn_ce1, AFI: l2vpn, SAFI: vpls-vpws
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 1
    Prefix                                       Next Hop                             MED         Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:300/112                                             0           100
-          Valid
Instance: l2vpn_ce2, AFI: l2vpn, SAFI: vpls-vpws
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 1
    Prefix                                       Next Hop                             MED         Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:310/112                                             0           100
-          Valid
Instance: l2vpn_ce3, AFI: l2vpn, SAFI: vpls-vpws
  Hostname: Local, Peer IP: 0.0.0.0
  Source IP: 0.0.0.0, Total routes: 1
    Prefix                                       Next Hop                                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:400/112                                                    0          100
-          Valid
Instance: default, AFI: l2vpn, SAFI: vpls
  Hostname: spine1, Peer IP: fe80::781a:c6ff:fec0:1
  Source IP: fe80::7857:d6ff:fec0:0, Total routes: 3
    Prefix                                       Next Hop                                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:301/112         192:1::1                                    0          -
4200000000 Valid
    00.00.00.00.00.00.00.00.00.00:311/112         192:1::1                                    0          -
4200000000 Valid
    00.00.00.00.00.00.00.00.00.00:401/112         192:1::1                                    0          -
4200000000 Valid
<...>
```

Example 2: Summary view of the BGP rib-in where you can find received information for address family l2vpn vpls.

```
supervisor@rtbrick: op> show bgp rib-in l2vpn vpls
Flags: & - Imported, ! - Error
Instance: default, AFI: l2vpn, SAFI: vpls
   Hostname: spine1, Peer IP: fe80::9a19:2cff:fe99:4701
   Source IP: fe80::9a19:2cff:fe36:3e03, Total routes: 3
    Prefix                                       Next Hop                                    MED        Lpref
AS Path    Status
    00.00.00.00.00.00.00.00.00.00:301/112         192:1::1                                    0          -
4200000000, 4200000003 Valid
    00.00.00.00.00.00.00.00.00.00:311/112         192:1::1                                    0          -
4200000000, 4200000003 Valid
    00.00.00.00.00.00.00.00.00.00:401/112         192:1::1                                    0          -
4200000000, 4200000003 Valid
```

Example 3: Summary view of the BGP rib-in with the error flag.

```
supervisor@rtbrick>rtbrick.net: op> show bgp rib-in ipv4 unicast peer address
12.0.0.2
Flags: & - Imported, ! - Error
Instance: default, AFI: ipv4, SAFI: unicast
  Hostname: P1, Peer IP: 12.0.0.2
  Source IP: 12.0.0.1, Total routes: 4
    Flags  Prefix           Next Hop   MED       Lpref       AS Path
           12.0.0.0/24      12.0.0.2   0         -           4200000002
           12.1.0.0/24      12.0.0.2   0         -           4200000002
           192.168.0.20/32  12.0.0.2   0         -           4200000002
           192.168.0.21/32  12.0.0.2   0         -           4200000002
```

BGP FIB

The 'show bgp fib' commands display the BGP forwarding table.

Syntax:

show bgp fib <option>

Option	Description
-	Without any option, the commands display the BGP forwarding table for all address families and all instances in a summary table format.
<afi>	BGP forwarding table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are l2vpn, 'ipv4' and 'ipv6'.
<afi> <safi>	BGP forwarding table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'unicast', 'labeled-unicast', 'vpn-multicast', 'vpn-unicast', 'evpn-vpws', and 'vpls-vpws'.
<afi> <safi> detail	Detailed list view of the BGP forwarding table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP forwarding table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP forwarding table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP forwarding table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP forwarding table entry for the given prefix and instance.

```
supervisor@rtbrick: op> show bgp fib
Instance: default, AFI: ipv6, SAFI: unicast
  Prefix                                          Preference      Out Label           Next Hop
  192:1::1/128                                    20              -                   fe80::781a:c6ff:fec0:1
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Prefix                                          Preference      Out Label           Next Hop
  192:1::1/128                                    20              -                   fe80::781a:c6ff:fec0:1
Instance: l2vpn_ce1, AFI: l2vpn, SAFI: vpls-vpws
  Prefix                                          Preference      Out Label           Next Hop
  00.00.00.00.00.00.00.00.00.00:301/112           20              150000,bos:1         192:1::1
Instance: l2vpn_ce2, AFI: l2vpn, SAFI: vpls-vpws
  Prefix                                          Preference      Out Label           Next Hop
  00.00.00.00.00.00.00.00.00.00:311/112           20              110000,bos:1         192:1::1
Instance: l2vpn_ce3, AFI: l2vpn, SAFI: vpls-vpws
  Prefix                                          Preference      Out Label           Next Hop
  00.00.00.00.00.00.00.00.00.00:401/112           20              120000,bos:1         192:1::1
```

BGP RIB-out

This command displays the routes that were advertised to peers.

Syntax:

show bgp rib-out <option>

Option	Description
-	Without any option, the command displays advertised BGP routes for all instances.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are l2vpn, 'ipv4' and 'ipv6'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are evpn, 'unicast', 'labeled-unicast', 'multicast', 'vpn-unicast', and vpls.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the routes advertised to a peer

```
supervisor@rtbrick: op> show bgp rib-out
Instance: default, AFI: ipv6, SAFI: unicast
  Peer-group: spine, Sent routes: 4
    Prefix                                     MED             Lpref             Origin          Next Hop
AS Path
    192:1::1/128                               0               -                 Incomplete      -
```

```
4200000003, 4200000000
    192:1::3/128                            0               -                 Incomplete      -
4200000003
    192:1::5/128                            0               -                 Incomplete      -
4200000003
    192:1::6/128                            0               -                 Incomplete      -
4200000003
Instance: default, AFI: ipv6, SAFI: labeled-unicast
  Peer-group: spine, Sent routes: 4
    Prefix                                  MED             Lpref             Origin          Next Hop
AS Path
    192:1::1/128                            0               -                 Incomplete      -
4200000003, 4200000000
    192:1::3/128                            0               -                 Incomplete      -
4200000003
    192:1::5/128                            0               -                 Incomplete      -
4200000003
    192:1::6/128                            0               -                 Incomplete      -
4200000003
Instance: default, AFI: l2vpn, SAFI: vpls
  Peer-group: spine, Sent routes: 3
    Prefix                                  MED             Lpref             Origin          Next Hop
AS Path
    00.00.00.00.00.00.00.00.00.00:300/112         0               -                 Incomplete      192:1::3
4200000003
    00.00.00.00.00.00.00.00.00.00:310/112         0               -                 Incomplete      192:1::3
4200000003, 4200000000
    00.00.00.00.00.00.00.00.00.00:400/112         0               -                 Incomplete      192:1::3
4200000003
```

BGP L2VPN

This command displays the BGP L2VPN information.

Syntax:

show bgp l2vpn <option>

Option	Description
pseudowire	Displays pseudowire peer IP information for all instances.
pseudowire instance <name>	Displays pseudowire peer IP information for the specified instance.

Example: Display L2VPN pseudowire information for instance l2vpn_ce1

```
supervisor@rtbrick: op> show bgp l2vpn pseudowire instance l2vpn_ce1
   Instance: l2vpn_ce1, AFI: l2vpn, SAFI: vpls-vpws
   Peer IP: 12.0.0.2
   Route Distinguisher: 192.1.6.3:65006
   Number of local interfaces: 1
    Interface name       Prefix                                    Status         Local SID  Remote SID  Pop
Label           Push Label
    ifl-0/1/7/100        00.00.00.00.00.00.00.00.00.00:300/112      up             300        301
label:150001,bos:1 label:150001
```

Example: Display L2VPN pseudowire information

```
supervisor@rtbrick.net: cfg> show bgp l2vpn pseudowire
Instance: vrf1-blue, AFI: l2vpn, SAFI: evpn-vpws
  Route Distinguisher: 192.168.3.10:65010
  Number of local interfaces: 4
    Interface name       Prefix                                     Status          Local SID  Remote SID Pop
Label          Push Label
    hostif-0/0/2/101     00.00.00.00.00.00.00.00.00.00:300/112      Up                    300         301
label:20018,bos:1  label:20017,bos:1
    hostif-0/0/2/201     00.00.00.00.00.00.00.00.00.00:310/112      Up                    310         311
label:20019,bos:1  label:20018,bos:1
    hostif-0/0/2/401     00.00.00.00.00.00.00.00.00.00:401/112      Up                    401         402 -
-
    hostif-0/0/3/401     00.00.00.00.00.00.00.00.00.00:402/112      Up                    402         401 -
-
```

4. Layer 3 Services

4.1. L3VPN

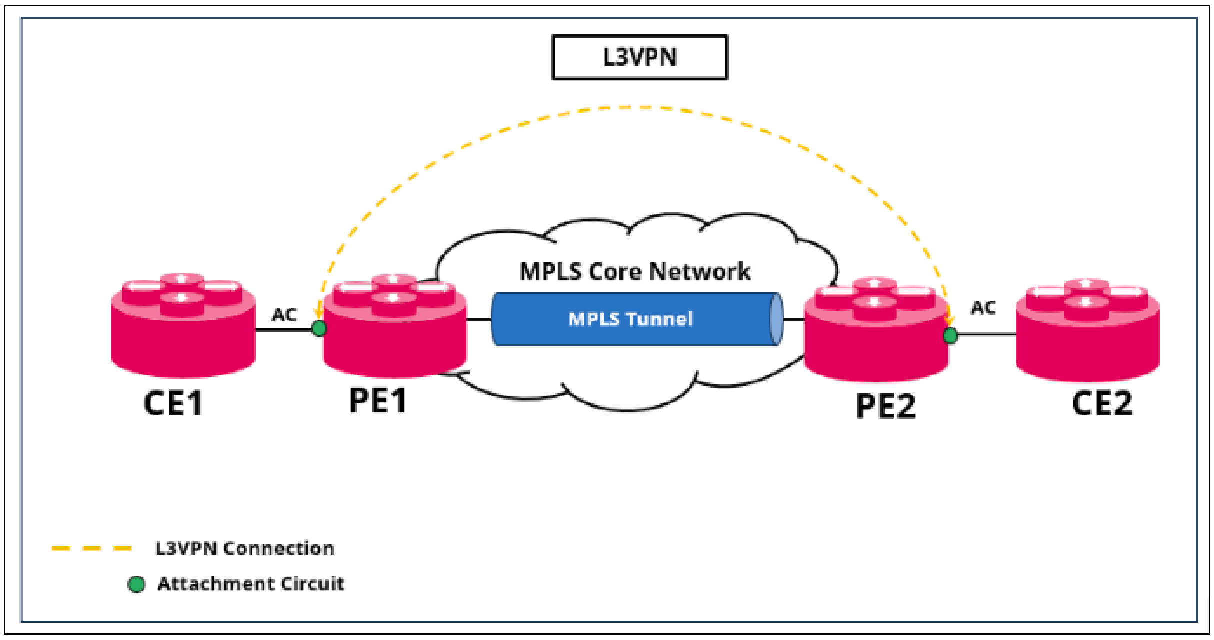
4.1.1. L3VPN Overview

L3VPN, or Layer 3 Virtual Private Network, is a type of MPLS VPN that operates on top of an IP/MPLS transport infrastructure. It allows customers to create secure and private layer 3(IP) connections over a shared infrastructure and provides seamless IP connectivity to the customer’s multiple sites.

The key aspects of L3VPN are:

- **IP-Based Routing**: All the control plane communication in the L3VPN network is based on IP-based routing protocols, so there are dynamic updates, multihoming, fault detection, and optimal data routing.
- **Provider Edge (PE)**: The Layer 3 device at the edge of the service provider network is called Provider Edge.
- **Customer Edge (CE)**: The Layer 3 device at the edge of the customer network with its interface connected to the service provider is called the Customer Edge. Both PE and CE have IP connectivity between them, either through Routing Protocols or Static Routing. The PE device has a separate instance configured for each customer. An Instance is a virtual routing and forwarding table (VRF) along with customer-private routing protocols. The CE device is responsible for transferring all the routing information in the customer sites to the PE.
- **Encapsulation and Tunneling**: L3VPN uses the MPLS data plane encapsulation to securely transfer data between two PE routers in the service provider network.
- **VPN Instances**: Each customer has a separate instance (VPN routing and forwarding (VRF)) in PE. This ensures that each customer’s data remains separate and maintains data privacy.
- **Security and Privacy**: L3VPN, using the above techniques, such as Encapsulation, Tunneling, and VPN instances, provides isolation for each customer in the shared network.
- **Scalability and Flexibility**: L3VPN supports different network deployments

depending on customer requirements. Instances are deployed only on the Edges, so they can be easily scaled. Filtering based on Extended Communities Route targets provides flexibility at the policy level, so it is possible to create various VPN topologies, such as hub-and-spoke VPNs, etc.



PE1 and PE2 are each configured with a BGP L3VPN instance connected to the customer edge devices (CEs). Both PE and CE have IP connectivity between them, either through Routing Protocols or Static Routing. Once PEs receive customer routes, they use Multi-Protocol BGP (MP-BGP) to exchange the routes between the PEs.

For example, when CE1 needs to transmit packets to CE2, then CE1 initiates a route lookup process to determine the nexthop for getting to CE2. This destination route is exchanged in advance between PEs and CEs through the configured routing protocol. Upon obtaining the routing destination via PEs, the CEs proceed to transmit the packet to PEs. After the PEs receive packets from the CEs, they add the VPN labels which are advertised by the neighboring PEs for the destination route, and send them to the peer PEs using the MPLS label-switched path transport (LSP). After the other PEs receive the packet via the MPLS LSP, they pop the VPN labels of the packets and perform an IP route lookup on the L3VPN instances, and forward the packets to the destination CE2.

Interprovider Layer 3 VPN options

Sometimes, a customer needs an L3 VPN between two locations where a given

Service Provider (SP) does not have coverage. To interconnect these customers, Layer 3 VPNs can be deployed in three different ways: Option A, Option B, and Option C.

- **Option A**: When dealing with another ISP, use eBGP, but place the BGP session within the customer's VRF. Essentially, treat the other service provider as if they were just another customer CE. This approach requires each customer's individual BGP and VRF configurations. However, it is ideal when you have no control over the other ISP's network.
- **Option B**: When setting up eBGP with the other ISP, make sure to use the global/default routing table. Each ASBR should advertise a unique label for every destination FEC in every VRF to the other ASBR. This process is similar to creating a label-switched path between the two ISPs.
- **Option C**: Configure eBGP between the ASBRs again, but this time using two planes: 1) The VPN unicast plane for exchanging VPN routes and plane 2 uses BGP Labeled Unicast, which enables PEs in one ISP to establish a label-switched path to the loopbacks of the PEs in the other ISP.

Supported L3VPN Standards

RFC Number	Description
RFC 4364	BGP/MPLS IP Virtual Private Networks (VPNs)

RFC and draft compliance are partial except as specified.

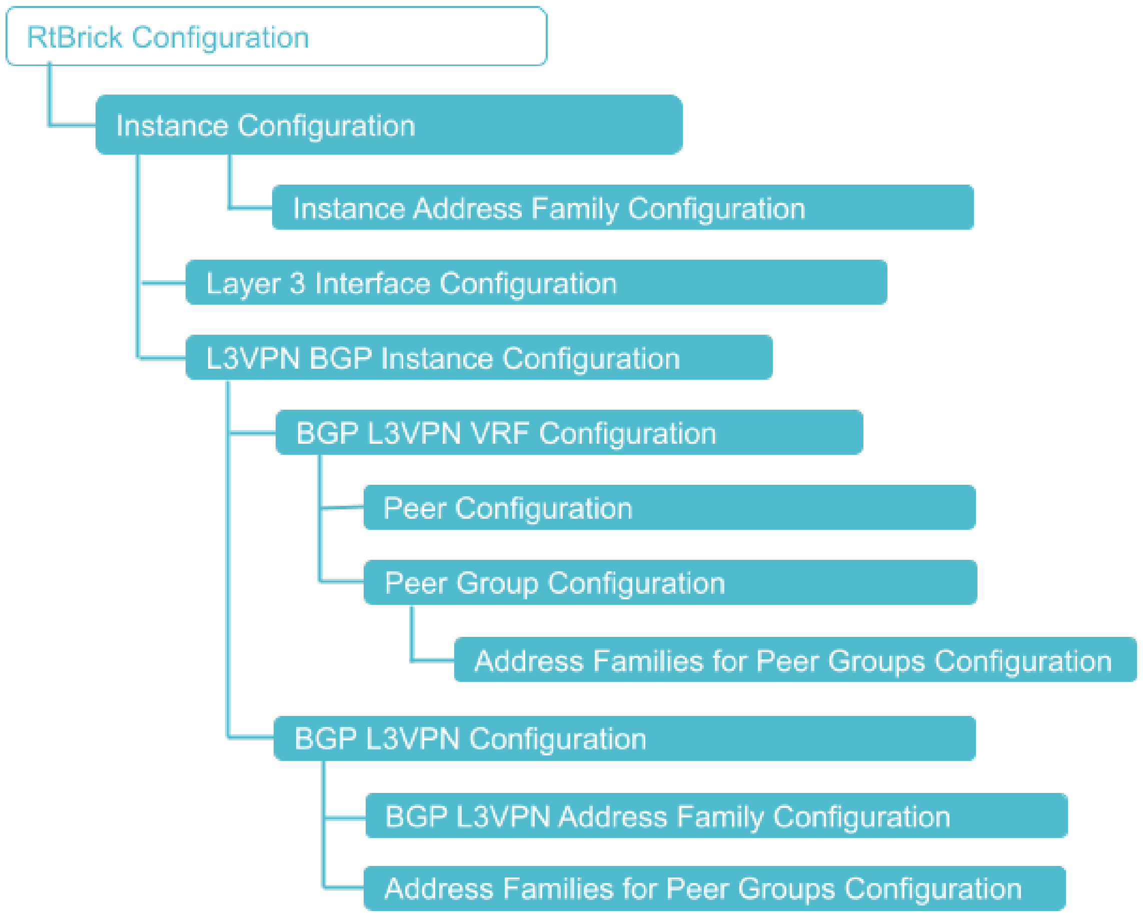
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

4.1.2. L3VPN Configuration

Configuration Hierarchy

The diagram below shows the L3VPN configuration hierarchy.


RtBrick Configuration
Instance Configuration
Instance Address Family Configuration
Layer 3 Interface Configuration
L3VPN BGP Instance Configuration
BGP L3VPN VRF Configuration
Peer Configuration
Peer Group Configuration
Address Families for Peer Groups Configuration
BGP L3VPN Configuration
BGP L3VPN Address Family Configuration
Address Families for Peer Groups Configuration

Configuration Syntax and Commands

The following sections describe the BGP configuration syntax and commands.

Instance Configuration

Each VPN is associated with one or more VPN routing and forwarding (VRF) instances. A VRF defines the VPN membership of a customer site attached to a PE router. A VRF consists of the following components:

- An IP version 4 (IPv4) 6 (IPv6) unicast routing table
- A private FIB table
- A set of interfaces that use the forwarding table
- A set of rules and routing protocol parameters that control the information that is included in the routing table

These components are collectively called a VRF instance.

The instance configuration hierarchy includes parameters that are required for or used by BGP, but that are not part of the BGP protocol configuration hierarchy itself. Route distinguishers and router IDs are configured directly at the instance hierarchy.

Syntax:

set instance <instance-name> <attribute> <value>

Attribute	Description
<instance-name>	Name of the routing instance
route-distinguisher <as-number|ipv4-address:id>	The route distinguisher (RD) uniquely defines routes within an IPv4 network. PE routers use route distinguishers to identify which VPN a packet belongs to. Supported formats are <as-number:id> or <ipv4-address:id>. If you want to use the format <as-number:id> with a 4-byte ASN, specify it with an "L". For example, set instance services route-distinguisher 4200000000L:101
ipv4-router-id <ipv4-address>	The router ID of the routing instance.

The following example configures tge instance identifier.

```
set instance l3vpn-ce1
set instance l3vpn-ce1 ipv4-router-id 172.16.3.10
set instance l3vpn-ce1 route-distinguisher 172.16.3.10:65001
commit
```

Example: Instance Identifier Configuration

```
supervisor@PE1: cfg> show config instance l3vpn-ce1
{
  "rtbrick-config:instance": [
    {
      "name": "l3vpn-ce1",
      "ipv4-router-id": "172.16.3.10",
      "route-distinguisher": "172.16.3.10:65001",
  <...>
```

Address Families

At the instance address family hierarchy, you can enable or disable address families for the instance, and configure parameters like route targets.

Please note default settings depend on the instance. For the 'default' instance, the IPv4 and IPv6 unicast, multicast, and labeled unicast, as well as the MPLS unicast address families are enabled by default. For any non-default instance, no address family is enabled by default and needs to be enabled by configuration.

Syntax:

```
set instance <instance-name> address-family <afi> <safi> <attribute> <value>
```

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast
route-target (import | export) <rt-value>	Route targets (RT) are used to transfer routes between VPN instances. The RT identifies a subset of routes that should be imported to or exported from a particular VPN instance. You can configure an RT for importing or exporting routes or both. If you want to use the format <as-number:id> with a 4-byte ASN, specify it with an "L". For example, set instance services address-family ipv4 unicast route-target export target:4200000000L:14

Attribute	Description
policy (import | export) <policy-name>	There are two attachment points for BGP policies. At this configuration hierarchy, you can attach import or export policies to the instance. These policies apply when routes are imported from the BGP protocol into the instance, or exported from the instance to the BGP protocol.

The following example configures the instance address families.

```
set instance l3vpn-ce1 address-family ipv4 unicast
set instance l3vpn-ce1 address-family ipv4 unicast route-target import
target:172.16.3.10:65001
set instance l3vpn-ce1 address-family ipv4 unicast route-target export
target:172.16.3.10:65001
set instance l3vpn-ce1 address-family ipv6 unicast
set instance l3vpn-ce1 address-family ipv6 unicast route-target import
target:172.16.3.11:65001
set instance l3vpn-ce1 address-family ipv6 unicast route-target export
target:172.16.3.11:65001
commit
```

Example: Instance Address Family Configuration

```
supervisor@PE1: cfg> show config instance l3vpn-ce1 address-family
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "unicast",
      "route-target": {
        "import": [
          "target:172.16.3.10:65001"
          ],
        "export": [
          "target:172.16.3.10:65001"
          ]
      }
    },
    {
      "afi": "ipv6",
      "safi": "unicast",
      "route-target": {
        "import": [
          "target:172.16.3.11:65001"
          ],
        "export": [
          "target:172.16.3.11:65001"
          ]
      }
    }
  ]
}
```

Layer 3 Interface Configuration

L3VPN supports the configuration of Layer 3 logical interfaces.

Syntax:

set interface <name> <attribute> <value>

Attribute	Description
unit <unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface.
inner-vlan <inner-vlan-id>	Inner VLAN ID.
instance <instance>	Assign the logical interface to an instance.
vlan <outer-vlan-id>	Outer VLAN ID.

The following example shows the interface configuration.

```
set interface ifp-0/0/2 unit 0
set interface ifp-0/0/2 unit 0 instance l3vpn-ce1
set interface ifp-0/0/2 unit 0 address ipv4 10.0.0.1/24
commit
```

Example: Interface Configuration

```
supervisor@PE1: cfg> show config interface ifp-0/0/2
{
  "rtbrick-config:interface": [
    {
      "name": "ifp-0/0/2",
      "unit": [
        {
          "unit-id": 0,
          "instance": "l3vpn-ce1",
          "address": {
            "ipv4": [
              {
                "prefix4": "10.0.0.1/24"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

Interface Address Configuration

This section describes how to configure interface IP addresses.

Syntax:

set interface <interface-name> **unit** <unit-id> **address** <afi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4 and ipv6
<prefix4|prefix6>	Assign IPv4 or IPv6 address to the interface unit.

The following show the interface address configuration.

```
set interface ifp-0/0/2 unit 0
set interface ifp-0/0/2 unit 0 instance l3vpn-ce1
set interface ifp-0/0/2 unit 0 address ipv4 10.0.0.1/24
commit
```

Example: Interface Address Configuration

```
supervisor@PE1: cfg> show config interface ifp-0/0/2
{
  "rtbrick-config:interface": [
    {
      "name": "ifp-0/0/2",
      "unit": [
        {
          "unit-id": 0,
          "instance": "l3vpn-ce1",
          "address": {
            "ipv4": [
              {
                "prefix4": "10.0.0.1/24"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

L3VPN BGP Instance Configuration

BGP L3VPN VFT (Virtual Forwarding Table) Configuration

Syntax:

set instance <name> **protocol bgp** <attribute> <value>

Attribute	Description
<name>	Name of the routing instance
host-name <host-name>	The name of the BGP host, to a maximum of 64 characters
domain-name <domain-name>	The name of the BGP routing domain, to a maximum of 64 characters
address-family <afi>	Address family identifier (AFI). Supported value: ipv4, ipv6
address-family <afi> <safi>	Subsequent address family identifier (SAFI). Supported value: unicast
local-as <as-number>	The AS number in four-byte format. The numbers allowed are from 1 to 4294967295.
med <med-value>	The BGP Multi-Exit Discriminator (MED) value. The numbers allowed are from 0 to 4294967295. When an AS has multiple links to another AS, the MED value is used to determine the exit to use to reach the other AS.
router-id <router-id>	Router identifier in IPv4 format
timer hold-time <seconds>	Hold timer in seconds. The valid range is 5 to 65535.
timer keepalive <seconds>	Keep a live timer in seconds. The valid range is 5 to 65535.
redistribute <source>	Enable the redistribution feature to dynamically inject specific types of routes into the BGP protocol. Supported route sources are direct, igmp, ipoe, isis, ospf, pim, ppp, static, and arp-nd.
redistribute <source> policy <policy>	Attach a policy to the redistribution process

The following example configures BGP L3VPN VFT (Virtual Forwarding Table).

```
set instance l3vpn-ce1 protocol bgp domain-name PE1-AS1
set instance l3vpn-ce1 protocol bgp hostname PE1-AS1
set instance l3vpn-ce1 protocol bgp local-as 65001
set instance l3vpn-ce1 protocol bgp router-id 172.16.3.10
set instance l3vpn-ce1 protocol bgp address-family ipv4 unicast
set instance l3vpn-ce1 protocol bgp address-family ipv6 unicast
commit
```

Example: BGP L3VPN VFT (Virtual Forwarding Table)

```
supervisor@PE1: cfg> show config instance l3vpn-ce1 protocol bgp
{
  "rtbrick-config:bgp": {
    "domain-name": "PE1-AS1",
    "hostname": "PE1-AS1",
    "local-as": 65001,
    "router-id": "172.16.3.10",
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "unicast"
      },
      {
        "afi": "ipv6",
        "safi": "unicast"
      }
    ],
 <...>
```

Peer Configuration

Once peer groups have been defined, BGP peers can be configured at the peer configuration hierarchy. A peer can be specified by address, or by interface when using IPv6 auto-discovered neighbors and link-local addresses. Furthermore, it is possible to configure TCP authentication and bind it to a peer.

Syntax to configure a BGP peer by address:

set instance <instance-name> **protocol bgp peer** (ipv4 | ipv6) <peer-address> <update-source> **peer-group** <peer-group>

Syntax to configure a BGP peer using IPv6 link-local addresses:

set instance <instance-name> **protocol bgp peer interface** <name> **peer-group** <peer-group>

Syntax to configure TCP Authentication for BGP peers:

set instance <instance-name> **protocol bgp peer** (ipv4 | ipv6) <peer-address> <update-source> **authentication-id** <authentication-id>

Attribute	Description
interface <name>	Enable BGP peer using IPv6 link-local addresses
ipv4 <peer-address>	IPv4 address of a BGP peer
ipv6 <peer-address>	IPv6 address of a BGP peer
allow-as-in <value>	Specify the value for allow-as-in. Allowed range of value 1 - 10.
<update-source>	Local IP address to be used for the peering
peer-group <peer-group>	Assign the peer to a peer group
authentication-id <authentication-id>	Authentication identifier

The following example configures a peer.

```
set instance l3vpn-ce1 protocol bgp peer ipv4 10.0.0.9 10.0.0.1
set instance l3vpn-ce1 protocol bgp peer ipv4 10.0.0.9 10.0.0.1 peer-group ce1
commit
```

Example: Peer Configuration

```
supervisor@PE1: cfg> show config instance l3vpn-ce1 protocol bgp peer
{
  "rtbrick-config:peer": {
    "ipv4": [
      {
        "peer-address": "10.0.0.9",
        "update-source": "10.0.0.1",
        "peer-group": "ce1"
      }
    ]
  }
}
```

Peer Group Configuration

In BGP, neighbor peers with the same update policies can be grouped to simplify the initial configuration and updates. Peers share the same policies such as route maps, distribution lists, filter lists, update sources, and so on, so peer groups only need one configuration statement for these values.

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> <attribute> <value>

Attribute	Description
local-as <as-number>	Local AS number for the peer group
remote-as <as-number>	Remote AS number for the peer group
any-as <true|false>	Enable dynamic AS negotiation for this peer group
ebgp-multihop <hop-count>	By default, the maximum number of hops between eBGP peers is 1 (direct connection). This hop count overrides the default behavior allowing connectivity between eBGP peers not directly connected.

The following example configures a peer group.

```
set instance l3vpn-ce1 protocol bgp peer-group ce1
set instance l3vpn-ce1 protocol bgp peer-group ce1 remote-as 65009
commit
```

Example: Peer Group Configuration

```
supervisor@PE1: cfg> show config instance l3vpn-ce1 protocol bgp peer-group
{
  "rtbrick-config:peer-group": [
    {
      "pg-name": "ce1",
      "remote-as": 65009,
<...>
```

Configuring Address Families for Peer Groups

At this configuration hierarchy, you can enable the address families that shall be supported for the group peers, and enable features specific to the address family.

Syntax:

set instance <instance-name> **protocol bgp peer-group** <peer-group-name> **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4, or ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: unicast
update-nexthop (ipv4-address | ipv6-address) <address>	BGP nexthop address for routes advertised to this peer group
policy (import | export) <policy-name>	Apply a routing policy to the peer group

The following example configures the address families that shall be supported for the group peers

```
set instance l3vpn-ce1 protocol bgp peer-group ce1 address-family ipv4 unicast
set instance l3vpn-ce1 protocol bgp peer-group ce1 address-family ipv6 unicast
commit
```

Example: Peer Groups Address Family Configuration

```
supervisor@PE1: cfg> show config instance l3vpn-ce1 protocol bgp peer-group ce1
address-family
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "unicast"
    },
    {
      "afi": "ipv6",
      "safi": "unicast"
    }
  ]
}
```

BGP L3VPN Configuration

Configuring the BGP L3VPN Address Families

Syntax:

set instance <name> **protocol bgp address-family** <afi> <safi>

Attribute	Description
<name>	Name of the routing instance
<afi>	Address family identifier (AFI). Supported values: ipv4, ipv6
<safi>	Subsequent address family identifier (SAFI). Supported values: vpn-unicast

To configure BGP L3VPN on the default instance, enter the following command:

```
set instance default protocol bgp address-family ipv4 vpn-unicast
set instance default protocol bgp address-family ipv6 vpn-unicast
commit
```

Example: BGP L3VPN Address Family Configuration

```
supervisor@PE1: cfg> show config instance default protocol bgp address-family
{
  "rtbrick-config:address-family": [
    {
      "afi": "ipv4",
      "safi": "vpn-unicast"
    },
    {
      "afi": "ipv6",
      "safi": "vpn-unicast"
    }
  ]
}
```

Configuring Address Families for Peer Groups

Syntax:

set instance <instance-name> **protocol bgp peer-group** <pg-name> **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported value: ipv4, ipv6.
<safi>	Subsequent address family identifier (SAFI). Supported value: vpn-unicast

Attribute	Description
add-path	Negotiate additional path capabilities with these peers, so that more than one path can be active to the peers in the group
default-information originate <true|false>	Generate and advertise a default route to peers in the group
extended-nexthop	Enable extended-next-hop encoding for BGP peer groups to allow the transfer of IPv4 prefixes over an IPv6 connection
nexthop-self <true|false>	Set the advertised BGP nexthop to yourself, this is the default for eBGP
nexthop-unchanged <true|false>	Do not modify the advertised BGP nexthop, this is the default for iBGP
update-nexthop (ipv4-address | ipv6-address) <address>	BGP nexthop address for routes advertised to this peer group
remove-private-as <true|false>	Remove private AS numbers from routes advertised to group peers
route-reflect-client <true|false>	Configure this peer as a route reflector client
policy (import | export) <policy-name>	Apply a routing policy to the peer group

To configure BGP L3VPN Peer Group on the default instance, enter the following command:

```
set instance default protocol bgp peer-group PE2-AS1 address-family ipv4 vpn-
unicast
set instance default protocol bgp peer-group PE2-AS1 address-family ipv6 vpn-
unicast
commit
```

Example: Configuring Address Families for Peer Groups

```
supervisor@PE1: cfg> show config instance default protocol bgp peer-group PE2-AS1
{
  "rtbrick-config:peer-group": [
    {
      "pg-name": "PE2-AS1",
      "remote-as": 4200000001,
      "address-family": [
        {
          "afi": "ipv4",
          "safi": "vpn-unicast"
        },
        {
          "afi": "ipv6",
          "safi": "vpn-unicast"
        }
      ]
    }
  ]
}
```

4.1.3. L3VPN Operational Commands

L3VPN Show Commands

BGP Summary

This command displays BGP protocol parameters like attributes or timers that are generic to the BGP instance.

Syntax:

show bgp summary <option>

Attribute	Description
Option	Description
-	Without any option, the command displays the information for all instances.

Example 1: BGP summary for L3VPN VRF instance

```
supervisor@rtbrick: op> show bgp summary instance l3vpn-ce1
Instance: l3vpn-ce1
  General information
    Hostname: PE1, Domain name: PE1
    Local AS: 65001, Version: 4
    Local preference: 100, eBGP Protocol preference: 20, iBGP Protocol preference: 200
    Router ID: 172.16.3.10, Cluster ID: 172.16.3.10
    MED: 0, Enforce first AS: True
    Address families configured:
      ipv4: unicast
      ipv6: unicast
  Capabilities
    Route refresh: True, AS4: True, Graceful restart: False,
  Best route selection
```

```
    Always compare MED: False, Ignore as path: False
    Ignore local preference: False, Ignore origin: False
    Ignore MED: False, Ignore route source: False
    Ignore router ID: False, Ignore uptime: True
    Ignore cluster length: False, Ignore peer IP: False
    Route select parameter: 0
  Timers
    Connect retry: 30s, Keepalive: 30s, Holdtime: 90s
  Statistics
    Peers configured: 1, Peers auto discovery: 0
      Peers in idle          : 0
      Peers in connect       : 0
      Peers in active        : 0
      Peers in opensent      : 0
      Peers in openconfirm   : 0
      Peers in established   : 1
```

Example 2: BGP summary for L3VPN default instance

```
supervisor@rtbrick: op> show bgp summary instance default
Instance: default
  General information
    Hostname: PE1, Domain name: PE1-BGP-Peers
    Local AS: 4200000001, Version: 4
    Local preference: 100, eBGP Protocol preference: 20, iBGP Protocol preference: 200
    Router ID: 192.168.0.20, Cluster ID: 192.168.0.20
    MED: 0, Enforce first AS: True
    Address families configured:
      ipv4: vpn-unicast
      ipv6: vpn-unicast
  Capabilities
    Route refresh: True, AS4: True, Graceful restart: False,
  Best route selection
    Always compare MED: False, Ignore as path: False
    Ignore local preference: False, Ignore origin: False
    Ignore MED: False, Ignore route source: False
    Ignore router ID: False, Ignore uptime: True
    Ignore cluster length: False, Ignore peer IP: False
    Route select parameter: 0
  Timers
    Connect retry: 30s, Keepalive: 30s, Holdtime: 90s
  Statistics
    Peers configured: 1, Peers auto discovery: 0
      Peers in idle          : 0
      Peers in connect       : 0
      Peers in active        : 0
      Peers in opensent      : 0
      Peers in openconfirm   : 0
      Peers in established   : 1
```

BGP Peer

The 'show bgp peer' commands display information on BGP peers.

Syntax:

show bgp peer <option>

Option	Description
-	Without any option, the commands display all BGP peers in all instances in a summary table format.
detail	Detailed information on all BGP peers in all instances in a list view.
<peer-name>	Detailed information on the peer with the given name.
address <peer-address>	Detailed information on the peer with the given IP address.
history	Displays BGP peer history information such as the peer state down reasons.
history <peer-address>	Displays BGP peer history information such as the peer state down reasons for a specified peer.
instance <instance-name>	Summary of all BGP peers in the given instance.
instance <instance-name> detail	Detailed information on all BGP peers in the given instance.
instance <instance-name> detail <peer-name>	Detailed information on the peer with the given name in the given instance.
instance <instance-name> detail address <peer-address>	Detailed information on the peer with the given IP address in the given instance.
statistics	Received and sent BGP prefixes per AFI/SAFI for all peers in all instances.
statistics peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name.
statistics peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address.
statistics instance <instance-name> peer <peer-name>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given name in the given instance.
statistics instance <instance-name> peer address <peer-address>	Received and sent BGP prefixes per AFI/SAFI for the peer with the given IP address in the given instance.

Example 1: BGP Peer Information for L3VPN VRF instance

```
supervisor@rtbrick: op> show bgp peer instance l3vpn-ce1
Instance: l3vpn-ce1
  Peer                                      Remote AS     State         Up/Down Time               PfxRcvd
PfxSent
  CE1                                       65009         Established   0d:07h:04m:09s             7
14
```

Example 2: BGP Peer Information for L3VPN default instance

```
supervisor@rtbrick: op> show bgp peer instance default
Instance: default
  Peer                                      Remote AS     State         Up/Down Time               PfxRcvd
PfxSent
  PE2                                       4200000002    Established   0d:07h:03m:03s             7
14
```

Example 3: Detailed view of BGP Peer Information for L3VPN VRF instance

```
supervisor@rtbrick: op> show bgp peer instance l3vpn-ce1 detail
Peer: CE1, Peer IP: 10.0.0.9, Remote AS: 65009, Local: 10.0.0.1, Local AS: 65001, Any AS: False
  Type: ebgp, State: Established, Up/Down Time: Thu Dec 05 05:22:33 GMT +0000 2024, Reason: Cease, Sub-Code:
Admin reset
  Discovered on interface: -
  Last transition: Thu Dec 05 05:22:33 GMT +0000 2024, Flap count: 1
  Peer ID        : 172.16.2.10, Local ID  : 172.16.3.10
  Instance       : l3vpn-ce1, Peer group: ce1
  6PE enabled    : False,
  TTL Security   : False, TTL Limit : -
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    Keepalive timer     : expires in 3s 266331us
    Holddown timer      : expires in 1m 26s 901069us
  NLRIs:
    Sent           : ['ipv4-unicast', 'ipv6-unicast']
    Received       : ['ipv4-unicast', 'ipv6-unicast']
    Negotiated     : ['ipv4-unicast', 'ipv6-unicast']
  Capabilities:
    Addpath sent                  : None
    Addpath received              : None
    Addpath negotiated            : None
    Extended nexthop sent         : None
    Extended nexthop received     : None
    Extended nexthop negotiated   : None
    Capabilities:
      Feature                      Sent            Received        Negotiated
      Route refresh                True            True            True
      4 byte AS                    True            True            True
      Graceful restart             False           False           False
      Link local only              False           False           False
  Prefix Limit:
  End of RIB:
    Address family                 Sent                                Received
    IPv4 unicast                   Thu Dec 05 05:22:38 GMT +0000 2024  Thu Dec 05 05:22:38 GMT +0000 2024
    IPv4 labeled-unicast           never                               never
    IPv6 unicast                   Thu Dec 05 05:22:38 GMT +0000 2024  Thu Dec 05 05:22:38 GMT +0000 2024
    IPv6 labeled-unicast           never                               never
    IPv4 VPN-unicast               never                               never
    IPv6 VPN-unicast               never                               never
    IPv4 flowspec                  never                               never
    IPv6 flowspec                  never                               never
    IPv4 VPN-multicast            never                             never
    L2VPN VPLS                    never                             never
    L2VPN EVPN                    never                             never
  RPKI Validation Stats:
    Address family                Valid    Invalid    Unknown
    IPv4 unicast                      0          0          0
    IPv6 unicast                      0          0          0
  Message stats:
    Session stats:
      Direction              Open          Update       Keepalive          Notify   Route refresh
      Input                     1               4            1035               0               0
      Output                    1               9            1032               0               0
    Total stats:
      Input                     2               8            1101               0               0
      Output                    3              11            1097               1               0
    Route stats:
      Address family                      Received            Sent    Prefix limit    Idle timeout
      IPv4 unicast                               4               4               0               0
      IPv4 labeled-unicast                       0               0               0               0
      IPv6 unicast                               3               3               0               0
      IPv6 labeled-unicast                       0               0               0               0
      IPv4 VPN-unicast                           0               0               0               0
      IPv6 VPN-unicast                           0               0               0               0
      IPv4 VPN-multicast                         0               0               0               0
      L2VPN VPLS                                 0               0               0               0
      L2VPN EVPN                                 0               0               0               0
      IPv4 flowspec                              0               0               0               0
      IPv6 flowspec                              0               0               0               0
```

Example 4: Detailed view of BGP Peer Information for L3VPN default instance

```
supervisor@rtbrick: op> show bgp peer instance default detail
Peer: PE2, Peer IP: 192.168.0.30, Remote AS: 4200000002, Local: 192.168.0.20, Local AS: 4200000001, Any AS:
False
  Type: ebgp, State: Established, Up/Down Time: Thu Dec 05 05:24:03 GMT +0000 2024
  Discovered on interface: -
  Last transition: Thu Dec 05 05:24:03 GMT +0000 2024, Flap count: 0
  Peer ID        : 192.168.0.30, Local ID  : 192.168.0.20
  Instance       : default, Peer group: PE2
  6PE enabled    : False,
  TTL Security   : False, TTL Limit : -
  Timer values:
    Peer keepalive : 30s, Local keepalive: 30s
    Peer holddown  : 90s, Local holddown : 90s
    Connect retry  : 30s
  Timers:
    Connect retry timer : 0s
    Keepalive timer     : expires in 15s 663139us
    Holddown timer      : expires in 1m 15s 207239us
  NLRIs:
    Sent          : ['ipv4-vpn-unicast', 'ipv6-vpn-unicast']
    Received      : ['ipv4-vpn-unicast', 'ipv6-vpn-unicast']
    Negotiated    : ['ipv4-vpn-unicast', 'ipv6-vpn-unicast']
  Capabilities:
    Addpath sent                : None
    Addpath received            : None
    Addpath negotiated          : None
    Extended nexthop sent       : None
    Extended nexthop received   : None
    Extended nexthop negotiated : None
    Capabilities:
      Feature                   Sent            Received        Negotiated
      Route refresh             True            True            True
      4 byte AS                 True            True            True
      Graceful restart          False           False           False
      Link local only           False           False           False
  Prefix Limit:
  End of RIB:
    Address family                Sent                              Received
    IPv4 unicast                  never                             never
    IPv4 labeled-unicast          never                             never
    IPv6 unicast                  never                             never
    IPv6 labeled-unicast          never                             never
    IPv4 VPN-unicast              Thu Dec 05 05:24:08 GMT +0000 2024  Thu Dec 05 05:24:08 GMT +0000 2024
    IPv6 VPN-unicast              Thu Dec 05 05:24:08 GMT +0000 2024  Thu Dec 05 05:24:08 GMT +0000 2024
    IPv4 flowspec                 never                               never
    IPv6 flowspec                 never                               never
    IPv4 VPN-multicast            never                               never
    L2VPN VPLS                    never                               never
    L2VPN EVPN                    never                               never
  Message stats:
    Session stats:
      Direction              Open          Update       Keepalive          Notify   Route refresh
      Input                     1               9            1033               0               0
      Output                    1               4            1035               0               0
    Total stats:
      Input                     1               9            1033               0               0
      Output                    2               4            1035               0               0
    Route stats:
      Address family                          Received            Sent    Prefix limit    Idle timeout
      IPv4 unicast                                   0               0               0               0
      IPv4 labeled-unicast                           0               0               0               0
      IPv6 unicast                                   0               0               0               0
      IPv6 labeled-unicast                           0               0               0               0
      IPv4 VPN-unicast                               4               4               0               0
      IPv6 VPN-unicast                               3               3               0               0
      IPv4 VPN-multicast                             0               0               0               0
      L2VPN VPLS                                     0               0               0               0
      L2VPN EVPN                                     0               0               0               0
      IPv4 flowspec                                  0               0               0               0
      IPv6 flowspec                                  0               0               0               0
```

BGP RIB-in

This command displays the received routes.

Syntax:

show bgp rib-in <option>

Option	Description
-	Without any option, the command displays information on the received BGP routing table on all instances in a summary table format.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'ipv4', 'ipv6' and 'l2vpn'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'evpn', 'labeled-unicast', 'unicast', 'vpn-multicast', 'vpn-unicast', and 'vpls'.

Option	Description
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> community <community-name>	BGP community details for the given AFI, SAFI, and instance.
<afi> <safi> error	BGP route with error status for the given AFI, SAFI, and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the BGP rib-in where you can find received information for the L3VPN VRF instance `l3vpn-ce1 ` of IPv4/IPv6 prefixes.

```
supervisor@rtbrick: op> show bgp rib-in ipv4 unicast
Flags: & - Imported, ! - Error, N - RPKI Unknown, I - RPKI Invalid, V - RPKI Valid
Instance: l3vpn-ce1, AFI: ipv4, SAFI: unicast
  Hostname: ce1, Peer IP: 10.0.0.9
  Source IP: 10.0.0.1, Total routes: 4
    Flags  Prefix                                       Next Hop                           MED         Lpref
AS Path
           10.0.0.0/24                                  10.0.0.9                           0           -
65009
           172.16.2.10/32                               10.0.0.9                           0           -
65009
           172.16.2.11/32                               10.0.0.9                           0           -
65009
           172.16.2.12/32                               10.0.0.9                           0           -
65009
```

```
supervisor@rtbrick: op> show bgp rib-in ipv6 unicast
Flags: & - Imported, ! - Error, N - RPKI Unknown, I - RPKI Invalid, V - RPKI Valid
Instance: l3vpn-ce1, AFI: ipv6, SAFI: unicast
  Hostname: ce1, Peer IP: 10.0.0.9
  Source IP: 10.0.0.1, Total routes: 3
    Flags  Prefix                                       Next Hop                           MED         Lpref
AS Path
           172:16:2::10/128                             10.0.0.9                           0           -
65009
```

```
          172:16:2::11/128                            10.0.0.9                           0           -
65009
          172:16:2::12/128                            10.0.0.9                           0           -
65009
supervisor@S1-STD-1-1011>bm13-tst.fsn.rtbrick.net: op>
```

Example 2: Summary view of the BGP rib-in where you can find received information of L3VPN default instance of address-family IPv4/IPv6 vpn-unicast

```
supervisor@rtbrick: op> show bgp rib-in ipv4 vpn-unicast
Flags: & - Imported, ! - Error, N - RPKI Unknown, I - RPKI Invalid, V - RPKI Valid
Instance: default, AFI: ipv4, SAFI: vpn-unicast
  Hostname: PE2, Peer IP: 192.168.0.30
  Source IP: 192.168.0.20, Total routes: 4
    Flags  Prefix                                     Next Hop                           MED         Lpref
AS Path
          10.0.1.0/24                                 192.168.0.30                       0           -
4200000002, 65029
          172.16.2.20/32                              192.168.0.30                       0           -
4200000002, 65029
          172.16.2.21/32                              192.168.0.30                       0           -
4200000002, 65029
          172.16.2.22/32                              192.168.0.30                       0           -
4200000002, 65029
```

```
supervisor@rtbrick: op> show bgp rib-in ipv6 vpn-unicast
Flags: & - Imported, ! - Error, N - RPKI Unknown, I - RPKI Invalid, V - RPKI Valid
Instance: default, AFI: ipv6, SAFI: vpn-unicast
  Hostname: PE2, Peer IP: 192.168.0.30
  Source IP: 192.168.0.20, Total routes: 3
    Flags  Prefix                                     Next Hop                           MED         Lpref
AS Path
          172:16:2::20/128                            192.168.0.30                       0           -
4200000002, 65029
          172:16:2::21/128                            192.168.0.30                       0           -
4200000002, 65029
          172:16:2::22/128                            192.168.0.30                       0           -
4200000002, 65029
```

BGP FIB

The 'show bgp fib' commands display the BGP forwarding table.

Syntax:

show bgp fib <option>

Option	Description
-	Without any option, the commands display the BGP forwarding table for all address families and all instances in a summary table format.

Option	Description
<afi>	BGP forwarding table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'l2vpn', 'ipv4' and 'ipv6'.
<afi> <safi>	BGP forwarding table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'unicast', 'labeled-unicast', 'vpn-multicast', 'vpn-unicast', 'evpn-vpws', and 'vpls-vpws'.
<afi> <safi> detail	Detailed list view of the BGP forwarding table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP forwarding table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP forwarding table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP forwarding table for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> <prefix>	BGP forwarding table entry for the given prefix and instance.

Example: Summary view of the BGP FIB

```
supervisor@rtbrick: op> show bgp fib
Instance: l3vpn-ce1, AFI: ipv4, SAFI: unicast, Total Routes: 8
  Prefix                                       Preference      Out Label            Next Hop
  10.0.0.0/24                                  20              -                    10.0.0.9
  10.0.1.0/24                                  20              20001,bos:1          192.168.0.30
  172.16.2.10/32                               20              -                    10.0.0.9
  172.16.2.11/32                               20              -                    10.0.0.9
  172.16.2.12/32                               20              -                    10.0.0.9
  172.16.2.20/32                               20              20001,bos:1          192.168.0.30
  172.16.2.21/32                               20              20001,bos:1          192.168.0.30
  172.16.2.22/32                               20              20001,bos:1          192.168.0.30
Instance: default, AFI: ipv4, SAFI: vpn-unicast, Total Routes: 8
  Prefix                                       Preference      Out Label            Next Hop
  10.0.0.0/24                                  20              20001,bos:1          10.0.0.9
  10.0.1.0/24                                  20              20005,bos:1          192.168.0.30
  172.16.2.10/32                               20              20001,bos:1          10.0.0.9
  172.16.2.11/32                               20              20001,bos:1          10.0.0.9
  172.16.2.12/32                               20              20001,bos:1          10.0.0.9
  172.16.2.20/32                               20              20005,bos:1          192.168.0.30
  172.16.2.21/32                               20              20005,bos:1          192.168.0.30
  172.16.2.22/32                               20              20005,bos:1          192.168.0.30
Instance: l3vpn-ce1, AFI: ipv6, SAFI: unicast, Total Routes: 6
  Prefix                                       Preference      Out Label            Next Hop
  172:16:2::10/128                             20              -                    10.0.0.9
```

```
   172:16:2::11/128                              20            -                    10.0.0.9
   172:16:2::12/128                              20            -                    10.0.0.9
   172:16:2::20/128                              20            20002,bos:1          192.168.0.30
   172:16:2::21/128                              20            20002,bos:1          192.168.0.30
   172:16:2::22/128                              20            20002,bos:1          192.168.0.30
 Instance: default, AFI: ipv6, SAFI: vpn-unicast, Total Routes: 6
   Prefix                                        Preference      Out Label            Next Hop
   172:16:2::10/128                              20            20002,bos:1          10.0.0.9
   172:16:2::11/128                              20            20002,bos:1          10.0.0.9
   172:16:2::12/128                              20            20002,bos:1          10.0.0.9
   172:16:2::20/128                              20            20006,bos:1          192.168.0.30
   172:16:2::21/128                              20            20006,bos:1          192.168.0.30
   172:16:2::22/128                              20            20006,bos:1          192.168.0.30
```

BGP RIB-out

This command displays the routes that were advertised to peers.

Syntax:

show bgp rib-out <option>

Option	Description
-	Without any option, the command displays advertised BGP routes for all instances.
<afi>	BGP routing table summary for the given address family (AFI), all sub-address families and all instances. Supported AFI values are 'l2vpn', 'ipv4' and 'ipv6'.
<afi> <safi>	BGP routing table summary for the given address family (AFI) and sub-address family (SAFI), and all instances. Supported SAFI values are 'evpn', 'unicast', 'labeled-unicast', 'multicast', 'vpn-unicast', and 'vpls'.
<afi> <safi> detail	Detailed list view of the BGP routing table for the given address family (AFI) and sub-address family (SAFI), and all instances.
<afi> <safi> <prefix>	BGP routing table entry for the given prefix and all instances.
<afi> <safi> instance <instance-name>	BGP routing table summary for the given AFI, SAFI, and instance.
<afi> <safi> instance <instance-name> detail	Detailed list view of BGP routing table for the given AFI, SAFI, and instance.

Option	Description
<afi> <safi> instance <instance-name> <prefix>	BGP routing table entry for the given prefix and instance.
<afi> <safi> peer <name> / peer address <ip>	Peer name or address

Example 1: Summary view of the routes advertised to a L3VPN VRF instance peer of IPv4/IPv6

```
supervisor@rtbrick: op> show bgp rib-out ipv4 unicast
Instance: l3vpn-ce1, AFI: ipv4, SAFI: unicast
  Peer-group: ce1, Sent routes: 8
    Prefix                                       MED     Lpref      Origin          Next Hop
      AS Path
    10.0.0.0/24                                  0       -          Incomplete      -
      [65001, 65009]
    10.0.1.0/24                                  0       -          Incomplete      -
      [65001, 4200000002, 65029]
    172.16.2.10/32                               0       -          Incomplete      -
      [65001, 65009]
    172.16.2.11/32                               0       -          Incomplete      -
      [65001, 65009]
    172.16.2.12/32                               0       -          Incomplete      -
      [65001, 65009]
    172.16.2.20/32                               0       -          Incomplete      -
      [65001, 4200000002, 65029]
    172.16.2.21/32                               0       -          Incomplete      -
      [65001, 4200000002, 65029]
    172.16.2.22/32                               0       -          Incomplete      -
      [65001, 4200000002, 65029]
```

```
supervisor@rtbrick: op> show bgp rib-out ipv6 unicast
Instance: l3vpn-ce1, AFI: ipv6, SAFI: unicast
  Peer-group: ce1, Sent routes: 6
    Prefix                                       MED     Lpref      Origin          Next Hop
      AS Path
    172:16:2::10/128                             0       -          Incomplete      -
      [65001, 65009]
    172:16:2::11/128                             0       -          Incomplete      -
      [65001, 65009]
    172:16:2::12/128                             0       -          Incomplete      -
      [65001, 65009]
    172:16:2::20/128                             0       -          Incomplete      -
      [65001, 4200000002, 65029]
    172:16:2::21/128                             0       -          Incomplete      -
      [65001, 4200000002, 65029]
    172:16:2::22/128                             0       -          Incomplete      -
      [65001, 4200000002, 65029]
```

Example 2: Summary view of the routes advertised to L3VPN default instance of address-family IPv4/IPv6 vpn-unicast

```
supervisor@rtbrick: op> show bgp rib-out ipv4 vpn-unicast
```

```
Instance: default, AFI: ipv4, SAFI: vpn-unicast
  Peer-group: PE2, Sent routes: 8
    Prefix                                      MED     Lpref      Origin          Next Hop
      AS Path
    10.0.0.0/24                                 0       -          Incomplete      -
      [4200000001, 65009]
    10.0.1.0/24                                 0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
    172.16.2.10/32                              0       -          Incomplete      -
      [4200000001, 65009]
    172.16.2.11/32                              0       -          Incomplete      -
      [4200000001, 65009]
    172.16.2.12/32                              0       -          Incomplete      -
      [4200000001, 65009]
    172.16.2.20/32                              0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
    172.16.2.21/32                              0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
    172.16.2.22/32                              0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
```

```
supervisor@rtbrick: op> show bgp rib-out ipv6 vpn-unicast
Instance: default, AFI: ipv6, SAFI: vpn-unicast
  Peer-group: PE2, Sent routes: 6
    Prefix                                      MED     Lpref      Origin          Next Hop
      AS Path
    172:16:2::10/128                            0       -          Incomplete      -
      [4200000001, 65009]
    172:16:2::11/128                            0       -          Incomplete      -
      [4200000001, 65009]
    172:16:2::12/128                            0       -          Incomplete      -
      [4200000001, 65009]
    172:16:2::20/128                            0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
    172:16:2::21/128                            0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
    172:16:2::22/128                            0       -          Incomplete      -
      [4200000001, 4200000002, 65029]
```

5. Subscriber Management

5.1. Subscriber Management

5.1.1. Subscriber Management Overview

RtBrick’s modular and scalable subscriber management, referred to as next-generation access infrastructure (ng-access), supports a range of access protocols including PPPoE, L2TPv2, IPoE (DHCPv4/v6), and RADIUS. This system is designed to deliver carrier-grade internet access services, with a focus on scalability, interoperability, and robustness. By providing support for multiple access protocols, RBFS ensures flexibility in deployment and compatibility with various network configurations, which is crucial for operators looking to provide consistent and high-quality services to their customers.

In this context, a subscriber refers to a customer connected through a Broadband Network Gateway (BNG). The BNG manages the connection by handling various protocols and sessions. Each subscriber is associated with a unique MAC address within a VLAN.

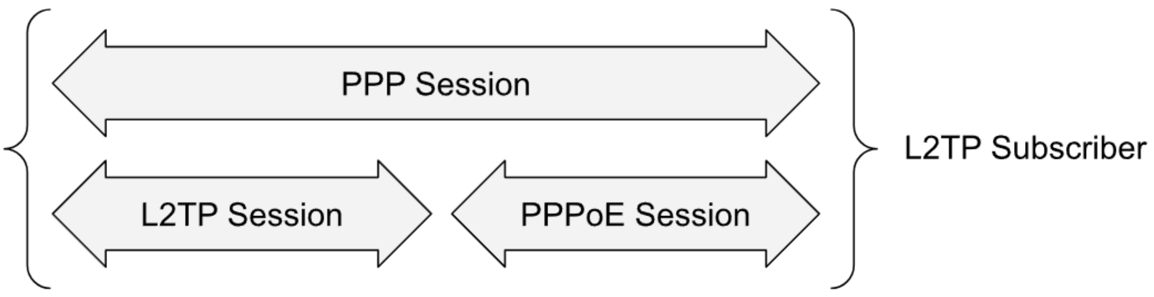
Providers that use separate VLANs for different services (e.g. internet and voice) may create multiple independent subscribers for the same customer or household. Although these subscribers might share certain QoS hierarchies, they are otherwise completely independent. Usually, VLANs used for services like IPTV or voice do not require subscriber management using static L3 interfaces with a traditional DHCP relay (non-subscriber).

A subscriber may have different forms of connectivity, depending on the access method used, such as PPPoE or IPoE. Thus, subscriber management involves overseeing customer connections, ensuring proper authentication, authorization, and accounting (AAA) of each subscriber, and maintaining the smooth operation of these sessions within a BNG. Effective subscriber management is key to maintaining high levels of customer satisfaction, as it directly impacts the reliability and performance of the internet services provided.

RBFS supports the following different protocol combinations to tailor an access network:

- **PPPoE Subscriber:** PPPoE/PPP Session and DHCPv6 (Prefix Delegation)

- **L2TP Subscriber:** PPPoE/PPP Session and L2TP Session
- **IPoE Subscriber:** DHCPv4 or DHCPv6



RtBrick’s implementation utilizes a distributed architecture to enhance both scalability and stability. The distributed nature of this architecture allows multiple processes to manage different aspects of subscriber connections concurrently, reducing bottlenecks and ensuring that the system can handle a large number of subscribers efficiently. Carrier networks often need to interoperate with a wide variety of client devices from different vendors, each requiring adherence to established industry standards and RFCs. This complexity necessitates a robust and flexible approach to managing different network interactions.

Furthermore, these networks must cope with the diverse behaviors of numerous access devices, some of which may have non-standard implementations but cannot be easily replaced due to logistical or cost-related constraints. This adds to the complexity of subscriber management, as non-standard devices may behave unpredictably, requiring additional handling to maintain stable connections. To address these challenges, RBFS provides a protocol stack specifically tailored to offer best-in-class interoperability, ensuring reliable connectivity across a diverse range of client devices. This approach not only ensures that the network can interoperate with devices from various vendors but also allows for customization and adaptation to unique network scenarios.

Overall, RtBrick’s subscriber management solution is designed to meet the needs of modern carrier networks by providing a flexible, scalable, and reliable platform for managing customer connections. Its distributed architecture, combined with support for a wide array of protocols and strong adherence to industry standards, makes it well-suited to handle the challenges posed by diverse client environments and evolving network requirements. This ensures that network operators can provide a seamless and high-quality experience to their customers, even in the face of varying device capabilities and non-standard behaviors.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Subscriber Management Daemons

There are four main daemons in the RtBrick distributed access architecture:

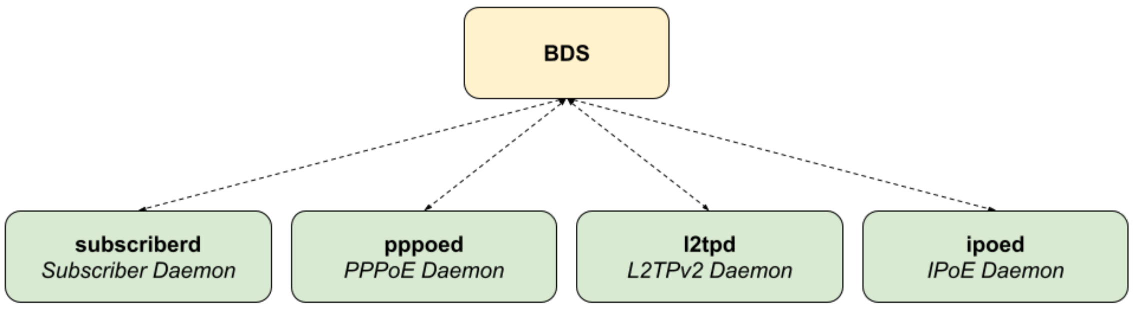


Figure 1. RBFS Access Infrastructure

The subscriber daemon (subscriberd) serves as the central component, managing the current subscriber state and handling Authentication, Authorization, and Accounting (AAA). The roles of each daemon are as follows:

- *subscriberd*: The subscriber daemon is responsible for subscriber management and AAA, which can be performed locally, via RADIUS, or through other methods. It also serves as the interface to the forwarding infrastructure, responsible to request per-subscriber QoS, filters, and more.
- *pppoed*: The PPPoE daemon handles PPPoE and PPP sessions for subscribers including subprotocols as ICMPv6 and DHCPv6 where the implemenation for PPPoE differs compared to IPoE.
- *l2tpd*: The L2TP daemon is responsible to manage L2TPv2 tunnels and sessions.
- *ipoed*: The IPoE (IP-over-Ethernet) daemon handles protocols related to IPoE subscribers like DHCPv4 and DHCPv6.

Subscriber Identifier

Each subscriber in RBFS is uniquely identified by a 64-bit number called subscriber-id, which remains unique across all RBFS processes and even across chassis in the context of high availability.

This uniqueness is achieved through the following format:

- The first 4 bits identify the chassis, starting from 0.
- The next 4 bits identify the process:

  0x00 subscriberd (e.g. L2BSA, test, …)

  0x01 pppoed (PPPoE sessions)

  0x02 l2tpd (reserved for LNS)

  0x03 ipoed (DHCPv4/v6)
- The following 8 bits are used to identify the application sharding instance. For example, instances such as pppoed.1, pppoed.2, and so on. While currently not in use, these bits provide the ability to partition load across multiple instances of the same process if needed to enhance scalability.
- The remaining 48 bits uniquely identify the subscriber, allowing for up to 281,474,976,710,656 unique subscribers per application instance before repetition, but even in the unlikely case of rollover, RBFS ensures uniqueness.

The subscriber portion always starts at '1', meaning that after a reboot, the subscriber identifiers will begin from the same value. These identifiers are consistent across multiple BNG, unless the chassis-id is explicitly set, which is required only for stateful high-availability configurations. Therefore, this number should not be used as a globally unique, non-repeating identifier in customer databases or management systems.

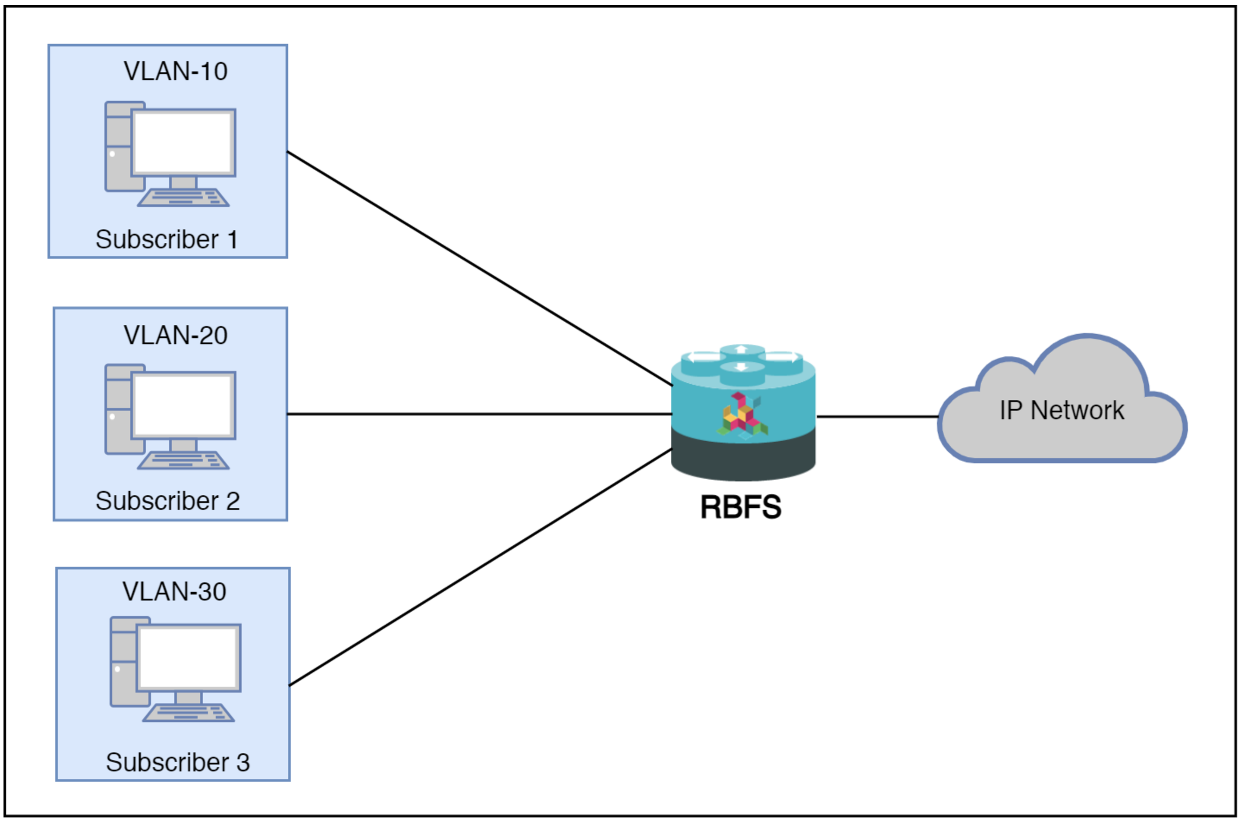
VLAN Modes and Encapsulations

RBFS supports both 1:1 VLAN (VLAN-per-subscriber) and N:1 VLAN (shared VLAN) models for PPPoE and IPoE subscribers. It supports various VLAN encapsulations. Subscriber VLANs are dynamically created by specifying VLAN ranges with corresponding configurations. After successful authentication, RBFS generates the subscriber IFL, which includes the dynamic VLAN.

1:1 VLAN Model (VLAN-per-subscriber)

In the 1:1 VLAN model, each subscriber is assigned a unique VLAN. This provides a direct one-to-one mapping between VLANs and subscribers. New services can be added easily without affecting others. The 1:1 model is recommended because it is simple and offers flexibility for features like backhauling, bitstream access, and QoS. As a result, more providers are migrating from the N:1 to the 1:1 VLAN model.

The following diagram illustrates a dedicated customer VLAN for each subscriber, showing the one VLAN per subscriber setup.



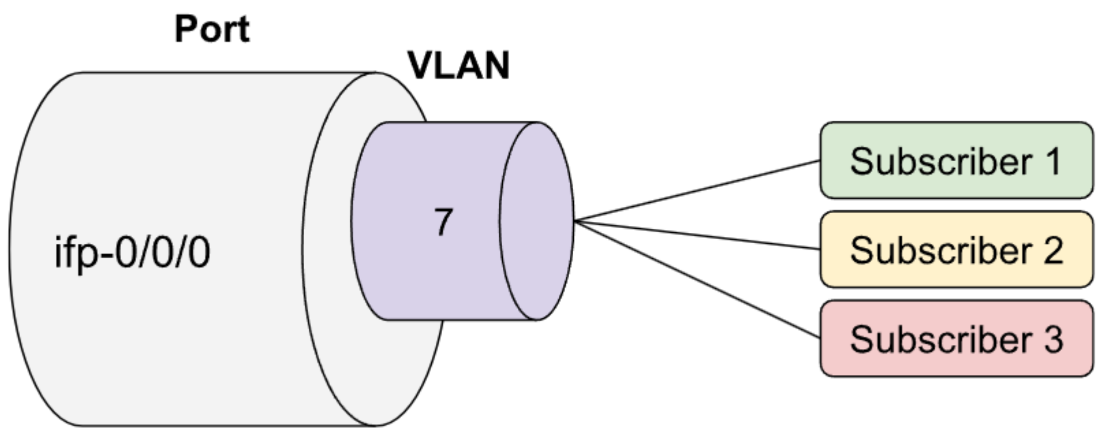
This model is defined by the maximum number of subscribers per VLAN in the access-interface configuration, which has a default value of 1. As a result, the 1:1 VLAN model is also the default in RBFS.

N:1 VLAN Model (shared-VLAN)

The shared VLAN model offers many-to-one (N:1) subscriber-to-service connectivity, where multiple subscribers share a single VLAN. Unlike the 1:1 model, where each subscriber has a dedicated VLAN, the N:1 model provides a shared VLAN for many subscribers. A drawback of the shared VLAN model is the lack of logical isolation between subscribers at the VLAN level.

This results in certain limitations for IPoE subscribers, such as the lack of support for features like HTTP redirect, subscriber ACL, NAT, and IGMP for IPoE N:1 subscribers.

The following diagram shows a single VLAN that is connected to many subscribers.



This model is defined by the maximum number of subscribers per VLAN in the access-interface configuration, which defaults to 1. Increasing this value beyond 1 enables the N:1 VLAN model for the corresponding access-interface.

Auto-sensing VLAN TPID

In modern telecommunications networks, managing traffic for wholesale partners presents challenges, especially when the network operator lacks knowledge of the specific TPID (Tag Protocol Identifier) used in arriving traffic. This can lead to inefficiencies in network handling and potential service disruptions. To address this issue, RBFS supports Auto-sensing VLAN TPID which automatically detects and processes traffic tagged with different TPIDs. Specifically, this functionality will support both TPID 0x8100, commonly referred to as 802.1q, and TPID 0x88a8, referred to as 802.1ad.

The auto-sensing mechanism will utilize the initial subscriber-initiated packet, such as PPPoE Discovery packets (PADI, PADR), DHCP Discovery, or DHCPv6 Solicit packets, to determine the appropriate TPID. This ensures that the system can adapt to the tagging method used by the incoming traffic without requiring prior knowledge.

The default behavior is auto-sensing the TPID based on the first incoming packet, offering maximum flexibility. However, network operators will still have the option to explicitly define the outer-vlan-encapsulation in the access-interface configuration if required, overriding the auto-sensing functionality when needed. This approach enhances the scalability and efficiency of handling wholesale traffic across various network environments.

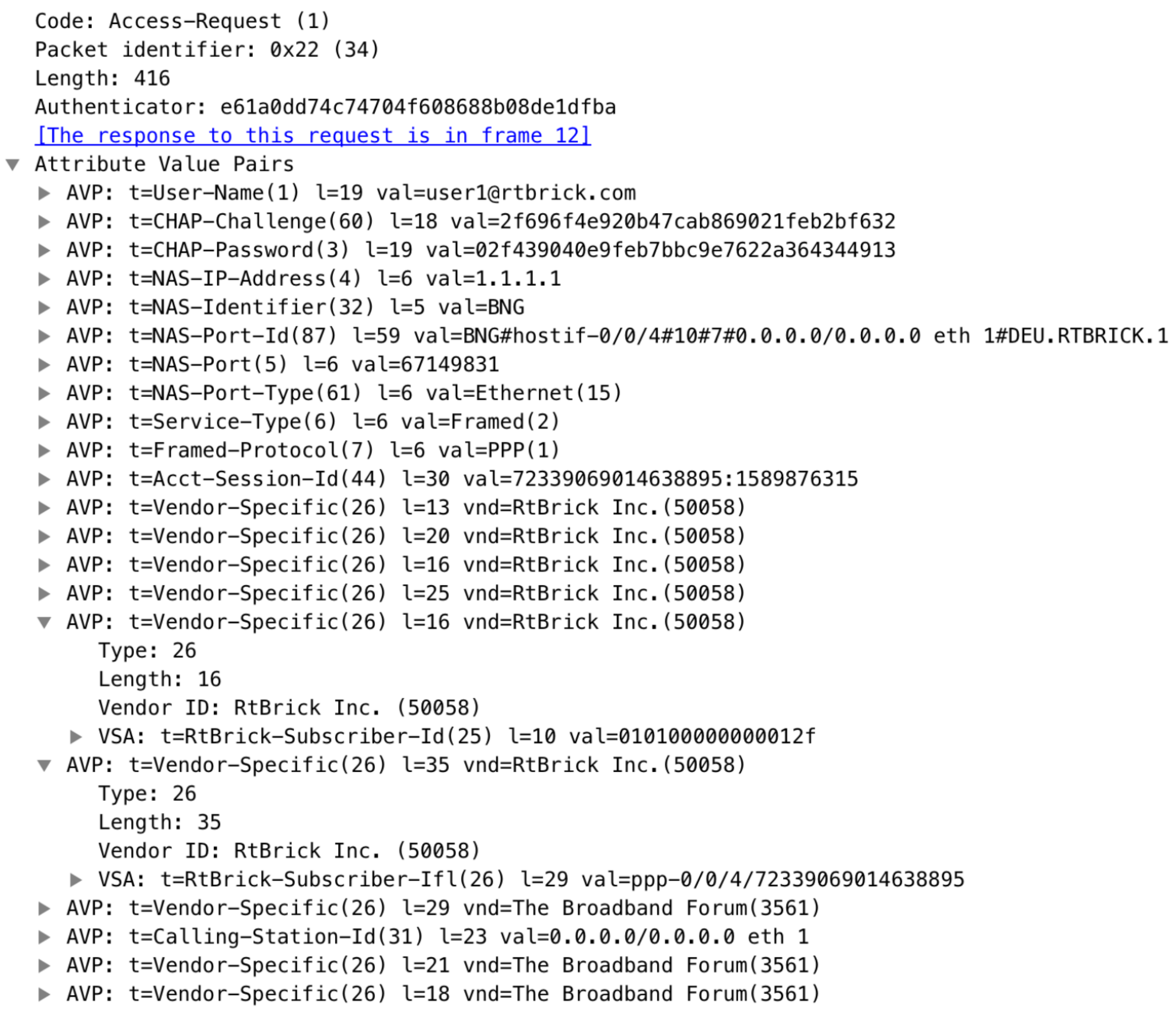
This applies to double-tagged profiles only! For single-tagged profiles, the default is 802.1q and cannot be changed, while for untagged profiles, no TPID check is performed as there is no VLAN present.

Authentication, Authorization, and Accounting Using RADIUS

Remote Authentication Dial-In User Service (RADIUS) is a protocol that provides centralized Authentication, Authorization, and Accounting (AAA) for all types of subscribers. RADIUS servers can perform as authentication and accounting servers. Authentication servers maintain authentication records for subscribers.

The subscriber daemon optionally requests authentication using RADIUS access request messages before permitting subscribers access. Accounting servers handle accounting records for subscribers. The subscriber daemon transmits RADIUS accounting-on, start, interim, and stop messages to the servers. Accounting is the process of tracking subscriber activity and network resource usage in a subscriber session. This includes the session time called time accounting and the number of packets and bytes transmitted during the session called volume accounting. A RADIUS server can behave as a change of authorization (CoA) client, allowing dynamic changes for subscriber sessions. The subscriber daemon supports both RADIUS CoA messages and disconnects messages. CoA messages can modify the characteristics of existing subscriber sessions without loss of service, disconnect messages can terminate subscriber sessions. Each RADIUS request from the subscriber daemon includes the RADIUS accounting-session-id attribute (type 44) with a format that is configurable in the AAA configuration profile and includes at least the subscriber-id to identify the corresponding subscriber. The default format (<subscriber-id>.<timestamp>) includes also a Unix timestamp to ensure that the tuple of NAS-Identifier (e.g. hostname) and Accounting-Session-Id is global and unique to be usable as a key in RADIUS databases.

Additionally, to subscriber-id and accounting-session-id each subscriber consists also of a subscriber-ifl build based on physical port information and subscriber-id (ifp: ifp-0/0/1 and subscriber-id: 72339069014638610 subscriber-ifl: ppp 0/0/1/72339069014638610) which is required as a handle in the RBFS forwarding infrastructure.



```
    Code: Access-Request (1)
    Packet identifier: 0x22 (34)
    Length: 416
    Authenticator: e61a0dd74c74704f608688b08de1dfba
    [The response to this request is in frame 12]
▼   Attribute Value Pairs
    ▶ AVP: t=User-Name(1) l=19 val=user1@rtbrick.com
    ▶ AVP: t=CHAP-Challenge(60) l=18 val=2f696f4e920b47cab869021feb2bf632
    ▶ AVP: t=CHAP-Password(3) l=19 val=02f439040e9feb7bbc9e7622a364344913
    ▶ AVP: t=NAS-IP-Address(4) l=6 val=1.1.1.1
    ▶ AVP: t=NAS-Identifier(32) l=5 val=BNG
    ▶ AVP: t=NAS-Port-Id(87) l=59 val=BNG#hostif-0/0/4#10#7#0.0.0.0/0.0.0.0 eth 1#DEU.RTBRICK.1
    ▶ AVP: t=NAS-Port(5) l=6 val=67149831
    ▶ AVP: t=NAS-Port-Type(61) l=6 val=Ethernet(15)
    ▶ AVP: t=Service-Type(6) l=6 val=Framed(2)
    ▶ AVP: t=Framed-Protocol(7) l=6 val=PPP(1)
    ▶ AVP: t=Acct-Session-Id(44) l=30 val=72339069014638895:1589876315
    ▶ AVP: t=Vendor-Specific(26) l=13 vnd=RtBrick Inc.(50058)
    ▶ AVP: t=Vendor-Specific(26) l=20 vnd=RtBrick Inc.(50058)
    ▶ AVP: t=Vendor-Specific(26) l=16 vnd=RtBrick Inc.(50058)
    ▶ AVP: t=Vendor-Specific(26) l=25 vnd=RtBrick Inc.(50058)
    ▼ AVP: t=Vendor-Specific(26) l=16 vnd=RtBrick Inc.(50058)
        Type: 26
        Length: 16
        Vendor ID: RtBrick Inc. (50058)
      ▶ VSA: t=RtBrick-Subscriber-Id(25) l=10 val=010100000000012f
    ▼ AVP: t=Vendor-Specific(26) l=35 vnd=RtBrick Inc.(50058)
        Type: 26
        Length: 35
        Vendor ID: RtBrick Inc. (50058)
      ▶ VSA: t=RtBrick-Subscriber-Ifl(26) l=29 val=ppp-0/0/4/72339069014638895
    ▶ AVP: t=Vendor-Specific(26) l=29 vnd=The Broadband Forum(3561)
    ▶ AVP: t=Calling-Station-Id(31) l=23 val=0.0.0.0/0.0.0.0 eth 1
    ▶ AVP: t=Vendor-Specific(26) l=21 vnd=The Broadband Forum(3561)
    ▶ AVP: t=Vendor-Specific(26) l=18 vnd=The Broadband Forum(3561)
```

Figure 2. RADIUS Access-Request

The subscriber-id is an unsigned 64-bit integer which is shown as a hex number in Wireshark.

Each subscriber is formed based on configuration profiles and individual settings retrieved via RADIUS. Conflicts between RADIUS-defined attributes and profile attributes are solved by prioritizing those received from RADIUS which is common best practice for broadband access concentrators. New subscribers are signalled via RADIUS access request and either accepted by RADIUS access-accept or rejected by RADIUS access-reject message from the RADIUS server. The RADIUS access-accept includes all attributes required to form the subscriber like IP addresses, DNS servers, and referenced configuration profiles. Some of those attributes can be changed by RADIUS dynamically using CoA requests without disconnecting the subscriber.

RADIUS Accounting

A RADIUS Acct-Status-Type attribute is used by the RADIUS client (subscriber daemon) to mark the start of accounting (for example, upon booting) by specifying Accounting-On and to mark the end of accounting (for example, just before a scheduled reboot) by specifying Accounting-Off. This message is often used by RADIUS servers to automatically close/terminate all open accounting records/sessions for the corresponding client, and therefore must not be sent to servers belonging to a profile that was already used/started for accounting.

Per default, the assumption is that all servers referenced by a RADIUS profile share the same states and therefore accounting-on must be only sent to one of those before the first accounting-start is sent.

RADIUS Accounting-On/Off messages are optionally enabled in the RADIUS Profile Configuration using the accounting-on-off attribute. The additional attribute accounting-on-wait prevents any new session until accounting has started meaning that the Accounting-On response is received.

Currently, Accounting-Off is not implemented.

RADIUS accounting requests are often used for billing and therefore should be able to store and retry over a more extended period (commonly up to 24 hours or more) which can be optionally enabled in the RADIUS profile configuration using the accounting-backup attribute. The maximum backup accounting hold time in seconds is defined in the attribute accounting-backup-max.

RADIUS Redundancy

RBFS allows configuring multiple RADIUS authentication and accounting servers for redundancy and or load-balancing. RADIUS redundancy ensures high availability in subscriber authentication, authorization, and accounting services. You can configure multiple RADIUS servers, and when one goes down, another takes over.

The following algorithms are supported:

- **DIRECT (default):** Requests are sent to the same server where the last request was sent. If the subscriber daemon receives no response from the server, requests are sent to the next server.
- **ROUND-ROBIN:** Requests are sent to servers in sequence, starting with the one following the server that handled the last request. If the subscriber

daemon router receives no response from the server, requests are sent to the next server.

RADIUS NAS Port Identifier

RADIUS NAS-Port-id is used to identify the port of a network access router that the subscriber management feature uses to authenticate subscribers.

The RADIUS attribute NAS-Port-Id (87) is formatted such as the following:

```
<NAS-IDENTIFIER>#<IFP>#<OUTER-VLAN>#<INNER-VLAN>#<ACI>#<ARI>
```

The Agent-Circuit-Id (ACI) and Agent-Remote-Id (ARI) are replaced with an empty string (##) if not available.

PPP over Ethernet (PPPoE)

PPP over Ethernet (PPPoE) is the common standard for internet access in the market.

PPPoE Session ID

The PPPoE session ID field is an unsigned 16-bit number as described in RFC2516. The values 0 and 65535 are reserved, with 0 used for PADI/PADO messages and 65535 reserved for future use. The session ID is unique within a broadcast domain (including IFP and VLANs) and for each subscriber device's MAC address. However, it is not unique per device or application instance. The session ID will change each time the session is reconnected.

PPPoE Service Name

When a subscriber device sends a PADI or PADR packet, the packet may include a service name field that specifies a particular service the client is requesting. Though, the system internally ignores the service name from the request, the name is still copied to the response packet sent back to the subscriber device.

If the request does not include any service name, the server's response (PADO or PADS) will include a default service name called access that ensures compatibility with certain clients, such as the Linux 'pppd'.

PPPoE AC-Cookie

This TAG is actually used to aid in protecting against denial-of-service attacks, but it is primarily used in RBFS to decide if a received PADR is a retry for an already answered (PADS send) one. The value itself is unpredictable and generated securely but it does not protect from reply attacks.

If a client receives this TAG in PADO, it MUST return the TAG unmodified in the following PADR. The TAG_VALUE is binary data of any value and length and is not interpreted by the Host.

The AC-Cookie is generated based on 8-bit salt followed by MD5 hash of salt, client MAC and dynamic PPPoE cookie secret.

```
    0                   1                   2
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | SALT           | MD5                             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The PPPoE cookie secret is randomly generated during the PPPoE daemon startup.

The AC-Cookie in the PADR creating the session is stored in the PPPoE PPP session object. For any received PADR it can be checked if there is a session on the same broadcast domain (IFP and VLAN’s) and MAC with the same AC-Cookie. In this case, the PADS is just retried.

If the broadcast domain and MAC is equal but AC-Cookie is different, this PADR must be considered as a new request.

This allows to separate two different PPPoE sessions on the same VLAN from the same MAC as frequently used by some service providers.

PPPoE Session Limit

A customer line is typically represented by one (single-tagged) or two VLAN (double-tagged) on a physical interface with a limitation to one session, which is also called the 1:1 VLAN mode.

In some cases, the customer CPE will set up multiple PPPoE sessions on a single VLAN which requires MAC limitations greater than one but less or equal to the per VLAN limitation.

Therefore RBFS supports two different session limitations in the access interface configuration (Access Interface Configuration), one per VLAN (max-subscribers-per-vlan) and an additional per client MAC address (max-subscribers-per-mac) both set to 1 per default as required for 1:1 VLAN mode.

The limitation of sessions per client MAC address must be less or equal the sessions per VLAN and the default set to one for both limits.

MTU Profiles

Bare metal switch hardware is typically limited in the number of supported MTU values. So RBFS has introduced the concept of MTU profiles with different types like physical, ip, IPoE, or pppoe. The last one is reserved for use with PPPoE sessions and applies to IPv4 and IPv6 traffic.

```
supervisor@switch: cfg> show config forwarding-options mtu-profile
{
  "rtbrick-config:mtu-profile": [
    {
      "mtu-profile-name": "IP-MTU-1500",
      "size": 1500,
      "type": "ip",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "IP-MTU-9202",
      "size": 9202,
      "type": "ip",
      "action": "drop"
    },
    {
      "mtu-profile-name": "__default_pppoe__",
      "size": 1492,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    }
  ]
}
```

The physical access interface should be configured with an MTU profile large enough to serve all IPoE and PPPoE MTU profiles, including the additional overhead for PPPoE and VLAN headers. Further details about interface MTU profiles can be found in the *Interfaces Configuration Guide*.

The Q2a/Q2c platform supports up to 8 MTU profiles in total including the default PPPoE profile default_pppoe. The default profile cannot be deleted but overwritten to change size and action.

The action could be either drop or redirect-to-cpu. The action drop silently discards all oversized packets. The action redirect-to-cpu punts oversized packets to the CPU where those could be either fragmented or dropped with ICMP response.

Fragmentation in software (cpu) must be explicitly enabled:

```
supervisor@switch: cfg> show config forwarding-options fragmentation
{
  "rtbrick-config:fragmentation": {
    "ipv4": {
      "state": "cpu"
    }
  }
}
```

MTU Profile Support for PPPeE Subscribers

The Point-to-Point Protocol (PPP) facilitates the negotiation of a Maximum Receive Unit (MRU) between endpoints, which is then applied as the Maximum Transmission Unit (MTU) on the opposite end of the PPP link. It is standard for each endpoint to independently negotiate its MRU, leading to potentially different MTU values for each direction of the PPP connection.

As an Internet Service Provider (ISP), control over the MTU/MRU sizes utilized for PPP over Ethernet (PPPoE) is not completely under your control. To address this, RBFS adopts a unique approach to managing PPPoE MTU profiles. Within RBFS, you can configure multiple PPPoE MTU profiles, however, they collectively count as a single entity against the eight MTU profile slots available on the Q2a/Q2c platform.

When a client initiates an MRU request via PPP Link Control Protocol (LCP) Configure-Request, RBFS employs this value to identify a matching MTU profile. This process involves sequentially comparing the requested MRU against an ordered list of MTU profiles until a profile with an MTU size equal to or greater than the requested MRU is located. If an exact match is found, the corresponding profile is selected, the exact match counter is incremented, and the MRU is deemed acceptable.

If no exact match exists, the system selects the closest, yet smaller, MTU profile from the list. This selection increments the best match counter, and the MTU from this profile is proposed to the client through a PPP LCP Configure-Nak message. Should the client reject the proposed MRU after three attempts, RBFS applies the default MTU profile *default_pppoe* as fallback MTU profile. This protocol ensures that the client's requested MRU is honored, but the transmission is governed by the largest available PPPoE MTU profile, thereby maximizing client compatibility.

For example, if a client requests an MRU of 1482 bytes but only profiles for 1472 and 1492 bytes are configured, the system will offer 1472 bytes as the best match through a PPP LCP Configure-Nak. The client may accept this offer by submitting a new PPP LCP Configure-Request for 1472 bytes or persist with the initial request of 1482 bytes. After three iterations without agreement, the fallback profile is activated, accepting the client's MRU of 1482 bytes but applying the default MTU of 1492 bytes. This approach is justified as most Customer Premises Equipment (CPE) devices support TCP Maximum Segment Size (MSS) clamping based on the negotiated MRU, ensuring that at least TCP traffic adheres to the negotiated limit. Furthermore, it is not unusual for CPEs to accept packets exceeding the negotiated MRU, which is why a larger MTU profile is designated as the fallback.

The negotiated MRU and applied MTU can be verified with the following command for every single PPPoE session.

```
user@switch: op> show pppoe session 72339069014638648 detail
Subscriber-Id: 72339069014638648
    State: ESTABLISHED
    ...
    PPP LCP:
        ...
        MRU: 1492 Peer: 1492
        MTU: 1492 Profile: __default_pppoe__
    ...
```

The command show pppoe mtu-profile lists all PPPoE MTU profiles in increasing order, with two statistics about the exact and best match.

```
supervisor@switch: op> show pppoe mtu-profile
Profile                 MTU      Exact Match      Best Match
PPPoE-MTU-1320          1320     0                0
PPPoE-MTU-1456          1456     0                0
PPPoE-MTU-1472          1472     0                0
__default_pppoe__       1492     0                0
```

The exact and best match counters can be used by operators to verify if the configured MTU profiles fit their environment or should be adopted.

As specified in RFC 2516, the Point-to-Point Protocol over Ethernet (PPPoE) requires a maximum negotiated MTU/MRU of 1492 bytes. However, RFC 4638 introduces the PPP-Max-Payload Tag, which extends the maximum negotiable MTU/MRU beyond 1492 bytes. This enhancement aims to reduce fragmentation within emerging broadband networks.

To configure an MTU/MRU exceeding 1492 bytes, a PPPoE client must incorporate the PPP-Max-Payload Tag within the PADI and PADR packets. Should the PPP-Max-Payload Tag specify a value greater than 1492 bytes, and if either the PPP LCP MRU or the MTU outlined in the access-profile is also configured above 1492, the tag will be replicated into the PADO and PADS packets issued by the RBFS. Echoing the PPP-Max-Payload Tag to the client signales support for MTU/MRU beyond 1492 bytes.

The value indicated by the PPP-Max-Payload Tag establishes the maximum threshold for both MTU and MRU settings.

Therefore, the highest permissible MTU/MRU is determined by the smaller of the two values, the configured PPP LCP MTU/MRU or the PPP-Max-Payload Tag's value.

At least one of the following values must be set to a value of more than 1492 bytes:

```
set access access-profile pppoe-dual protocol ppp lcp mru 1500
set access access-profile pppoe-dual protocol ppp lcp mtu 1500
```

It is also required to set an MTU profile with the desired MTU size:

```
set forwarding-options mtu-profile pppoe-1500
set forwarding-options mtu-profile pppoe-1500 size 1500
set forwarding-options mtu-profile pppoe-1500 type pppoe
set forwarding-options mtu-profile pppoe-1500 action redirect-to-cpu
```

For L2TPv2 tunneled PPPoE sessions, the MTU is enforced by the LNS. It's usual behavior that the LNS renegotiates the MTU. So LAC may not know the actual MTU. This is the reason why RBFS does not apply an MTU profile for such sessions.

RBFS overwrites the selected MTU profile with *default_l2tp* for L2TPv2 sessions. This profile must be explicitly created, otherwise it is ignored. The action must be drop because ICMP or fragmentation is not supported for tunneled sessions.

```
supervisor@switch: cfg> show config forwarding-options mtu-profile
__default_l2tp__
{
    "rtbrick-config:mtu-profile": [
      {
        "mtu-profile-name": "__default_l2tp__",
        "size": 1492,
        "type": "pppoe",
        "action": "drop"
      }
    ]
  }
```

This configuration allows to optionally enforce an MTU on LAC if needed.

Following an example MTU profile configuariton for PPPoE supporting all MTU values between the minimum of 1280 and maximum of 1500 in steps of 8 bytes.

```
supervisor@switch: cfg> show config forwarding-options mtu-profile
{
  "rtbrick-config:mtu-profile": [
    {
      "mtu-profile-name": "__default_pppoe__",
      "size": 1492,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1280",
      "size": 1280,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1288",
      "size": 1288,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1296",
      "size": 1296,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1304",
      "size": 1304,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1312",
      "size": 1312,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1320",
      "size": 1320,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1328",
      "size": 1328,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1336",
      "size": 1336,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1344",
      "size": 1344,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1352",
      "size": 1352,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1360",
      "size": 1360,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1368",
      "size": 1368,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1376",
      "size": 1376,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1384",
      "size": 1384,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1392",
      "size": 1392,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1400",
      "size": 1400,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1408",
      "size": 1408,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1416",
      "size": 1416,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1424",
      "size": 1424,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1432",
      "size": 1432,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1440",
      "size": 1440,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1448",
      "size": 1448,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1456",
      "size": 1456,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1464",
      "size": 1464,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1472",
      "size": 1472,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1480",
      "size": 1480,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1488",
      "size": 1488,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1490",
      "size": 1490,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    },
    {
      "mtu-profile-name": "PPPoE-MTU-1500",
      "size": 1500,
      "type": "pppoe",
      "action": "redirect-to-cpu"
    }
  ]
}
```

MTU Profile Support for IPoE Subscribers

Unlike the PPPoE protocol, which can negotiate MTU dynamically, IPoE relies on a fixed MTU setting. The MTU value is predefined and remains constant for all IPoE subscribers using the configured access interface.

For IPoE subscribers, MTU is statically configured and is applied at the access interface level.

Configuration commands for the MTU profile for IPoE subscribers.

```
set access access-profile ipoe
set access access-profile ipoe instance default
set access access-profile ipoe protocol dhcp enable true
set access access-profile ipoe protocol dhcp lease-time 3600
set access access-profile ipoe protocol dhcpv6 enable true
set access access-profile ipoe protocol dhcpv6 lifetime 3600
set access access-profile ipoe protocol ipoe ip-mtu-profile ipoe_mtu_1500
```

It is also required to set an MTU profile with the desired MTU size:

```
set forwarding-options mtu-profile ipoe_mtu_1500
set forwarding-options mtu-profile ipoe_mtu_1500 size 1500
set forwarding-options mtu-profile ipoe_mtu_1500 type ip
set forwarding-options mtu-profile ipoe_mtu_1500 action drop
```

Following is an example of access profile IPoE:

```
supervisor@rtbrick.net: cfg> show config access access-profile ipoe
```

```
{
  "rtbrick-config:access-profile": [
    {
      "profile-name": "ipoe",
      "instance": "default",
      "protocol": {
        "dhcp": {
          "enable": "true",
          "lease-time": 3600
        },
        "dhcpv6": {
          "enable": "true",
          "lifetime": 3600
        },
        "ipoe": {
          "ip-mtu-profile": "ipoe_mtu_1500"
        }
      },
      "address-family": {
        "ipv4": {
          "enable": "true",
          "pool-name": "poolv4"
        },
        "ipv6": {
          "enable": "true",
          "pool-name": "poolv6",
          "prefix-delegation-pool-name": "poolv6pd"
        }
      }
    }
  ]
}
```

Following an example MTU profile configuariton for IPoE:

```
supervisor@DT-STD-23-2401>bm13-tst.fsn.rtbrick.net: cfg> show config forwarding-
options mtu-profile
{
  "rtbrick-config:mtu-profile": [
    {
      "mtu-profile-name": "ipoe_mtu_1500",
      "size": 1500,
      "type": "ip",
      "action": "drop"
    }
  ]
}
```

PPPoE VLAN Profiles

This chapter describes the VLAN profile feature. If enabled for the access interface, then incoming sessions (e.g. PPPoE PADI/PADR) are not honored unless matching vlan-profile is found.

The VLAN profiles must be added to the table global.vlan.profile owned by PPPoE daemon. All entries in this table are ephemeral and therefore lost after reboot or PPPoE daemon restart.

Example:

```
{
    "table": {
        "table_name": "global.vlan.profile"
    },
    "objects": [
        {
            "attribute": {
                "ifp_name": "ifp-0/1/2",
                "outer_vlan_min": 128,
                "outer_vlan_max": 128,
                "inner_vlan_min": 1,
                "inner_vlan_max": 4095,
                "access_profile_name": "access-profile-vlan"
            }
        }
    ]
}
```

PPPoE Dual-Stack IPv4/IPv6 with DHCPv6

The whole IPv6 control plane of a PPPoE session like ICMPv6 router-solicitation (RS), ICMPv6 router-advertisement (RA) and DHCPv6 is handled in the PPPoE daemon (pppoed).

The PPPoE daemon handles received router solicitations by responding with router advertisements and is sending frequent router advertisements based on configured intervals.

The other-config flag in the router-advertisement is automatically set if DHCPv6 is enabled for this particular subscriber. This flag signals that there is more information available via DHCPv6.

DHCPv6 over PPPoE is different to DHCPv6 over Ethernet because of the special characteristics of point-to-point protocols. DHCPv6 over PPPoE is supporting delegated IPv6 prefixes (IA_PD) and DNS options only. Unsupported IA options (IA_NA and IA_TA) or options that can be served will be rejected with status code options as defined per RFC.

The delegated IPv6 prefix served by DHCPv6 will be assigned to the subscriber via RADIUS or local pool regardless of the protocols negotiated with the client.

DHCPv6 was primarily designed for use in Ethernet networks. The fact that Ethernet is connectionless requires that DHCPv6 servers must manage releases for the clients and free them automatically if a lease expires. Such extensive release management is not needed for connection-oriented protocols like PPPoE where addresses are assigned to the PPPoE session. This fact allows to implementing DHCPv6 nearly stateless on the server side by just tracking if an assigned prefix is assigned or released. This is tracked in the attribute ipv6pd_negotiated of the PPPoED/SubscriberD (global.ppp.1.subscriber.result) result object and copied to the actual subscriber object (local.access.subscriber). As this use case is covered by PPPoE, there is no lease expiry implemented.

The delegated-prefix is added to the subscriber-ifl only if negotiated and removed if not negotiated. The presence of delegated prefix in the subscriber-ifl is used by IFMD to add or remove the forwarding entry.

If DHCPv6 is enabled but no delegated prefix provided, only DNS is served in response if available.

PPPoE DHCPv6 Server DUID

The DHCPv6 server identifier DUID is generated based on IP6CP negotiated interface-identifier as shown below:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     DUID-Type 3 (DUID-LL)     |    hardware type 27 (EUI64)   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      interface-identifier                     |
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Layer Two Tunneling Protocol (L2TPv2)

This chapter describes the RtBrick Layer Two Tunneling Protocol (L2TPv2) implementation. This document describes also the corresponding configuration and operations commands for PPPoE access services with PPP tunneling using the Layer Two Tunneling Protocol version 2 (L2TPv2) on RtBRick FullStack (RBFS).

Typically, a user obtains a Layer 2 (L2) point-to-point connection to a Broadband Network Gateway (BNG) using the PPPoE protocol as described in RFC 2516 and runs PPP over that connection. In the most common case, the L2 termination point and PPP session endpoint reside on the same physical device. Tunneling protocols, such as L2TPv2 provide a dynamic mechanism for extending PPP by allowing the L2 and PPP endpoints reside on different devices that are interconnected by an IP network. This separation allows the actual processing of PPP packets to be divorced from the termination of the L2 circuit. The L2TP access concentrator (LAC) physically terminates the L2 connection and tunnels the PPP packets across an IP network to the L2TP network server (LNS). The LNS then terminates the logical PPP connection.

RFC and draft compliance are partial except as specified.

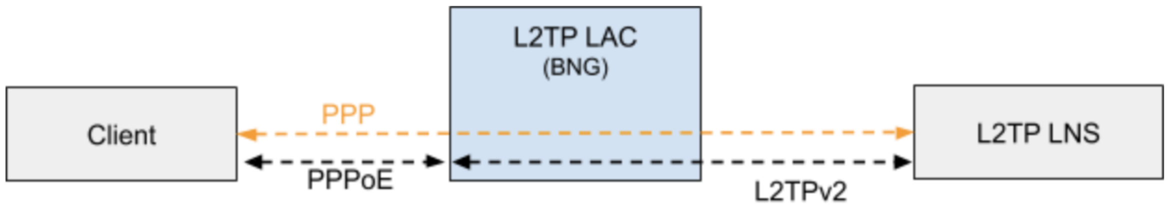


Figure 3. L2TP PPPoE

To establish a PPPoE session via L2TP, the tunnel-type must be configured as L2TP. This configuration can be achieved either for local users or by utilizing the corresponding tunnel-type attribute through RADIUS.

```
# Local User
cfg> set access user-profile local@l2tp tunnel-type L2TP

# FreeRADIUS
"radius@l2tp" Cleartext-Password := "test"
    Service-Type = Framed-User,
    Framed-Protocol = PPP,
    Tunnel-Type:0 = L2TP
```

Defining an L2TP configuration profile is essential, which can be referenced through an access-profile or by employing the appropriate RADIUS VSA. The actual tunnels may either be defined locally via an L2TP pool configuration or set up dynamically through RADIUS.

Currently, RBFS supports L2TP subscriber sessions with EtherType 0x8100 (802.1Q) only; it does not support EtherType 0x88a8 (802.1ad).

L2TP LAC

The L2TP Access Concentrator (LAC) is a node that acts as one side of an L2TP tunnel endpoint, and is a peer to the L2TP Network Server L2TP LNS. The LAC sits between a LNS and a remote system, and forwards packets to and from each.

L2TP LNS

The L2TP Network Server (LNS) is a node that acts as one side of an L2TP tunnel endpoint and is a peer to the L2TP Access Concentrator L2TP LAC. The LNS is the logical termination point of a PPP session that is being tunneled from the remote system by the LAC.

The LNS role is currently not supported!

L2TP Tunnel Selection

Each new session creates a session request object (local.l2tp.session.request) to track the tunnel selection progress, the currently selected ones, and which are already tried. This object is automatically deleted if the session setup is successful.

All tunnels in state DEAD are skipped in the tunnel selection but considered at the end if no other tunnels are available. Tunnels with a session limit reached are not considered for further sessions. To select a tunnel, the L2TP daemon first generates list of preferred tunnels based on tunnel preference, where the lowest value has the highest priority. The configured L2TP tunnel selection algorithm decides how to select a tunnel out of the remaining tunnels with the same preference. The RANDOM algorithm selects the tunnel randomly whereas BALANCED selects the least filled tunnel based on the number of sessions.

Following the L2TP tunnel pool order/priority in case, there are multiple pools available for a single subscriber:

1. RADIUS defined tunnel (RFC2866)
2. RADIUS VSA (RtBrick-L2TP-Pool) or local user profile
3. L2TP configuration profile

L2TP Control Channel

The control channel is responsible for the orderly passing of control messages between the tunnel endpoints and acts as a transport layer for reliable delivery of control messages and tunnel keep alive services for the tunnel.

Each L2TP tunnel is split into the actual tunnel object with all the information exchanged during tunnel establishment plus the FSM state and a separate control channel with the sequence numbers, window size, and thresholds changed with every sent and received packet.

RBFS sent a ZLB ACK only if there are no further messages waiting in queue for that peer, as well as to acknowledge multiple packets at once.

The HELLO keep-alive messages are also part of the control channel and only send if there is no other message sent if the queue is empty and no other message send during the hello interval.

L2TP Access Line Information (RFC5515)

Connect-Speed-Update-Notification (CSUN)

The Connect-Speed-Update-Notification (CSUN) is an L2TP control message sent by the LAC to the LNS to provide transmit and receive connection speed updates for one or more sessions.

This implementation will send one CSUN request per session!

CSUN requests are disabled per default and can be enabled in the L2TP profile L2TP Profile Configuration.

CSUN messages are defined in RFC5515, which is not widely supported. Therefore those messages are marked as not mandatory in RBFS to allow interwork with LNS servers not supporting RFC5515.

RFC2661:

```
The Mandatory (M) bit within the Message Type AVP has special
meaning. Rather than an indication of whether the AVP itself
should be ignored if not recognized, it is an indication as to
whether the control message itself should be ignored. Thus, if the
M-bit is set within the Message Type AVP and the Message Type is
unknown to the implementation, the tunnel MUST be cleared.  If the
M-bit is not set, then the implementation may ignore an unknown
message type.
```

RFC and draft compliance are partial except as specified.

Connect-Speed-Update-Request (CSURQ)

The Connect-Speed-Update-Request (CSURQ) is an L2TP control message sent by the LNS to the LAC to request the current transmission and receive connection speed for one or more sessions.

Sending or responding to CSURQ requests is currently not supported!

Access Line Information L2TP Attribute Value Pair Extensions

The corresponding access line information for a subscriber is included in the ICRQ message as defined in RFC5515.

Connect Speed Values

The default value for TX and RX Connect Speed is set to 1000000000 (1G) which is replaced by the actual data rate upstream/downstream of the corresponding access line information object or directly set using the RADIUS attributes RtBrick-L2TP-Tx-Connect-Speed (42) and RtBrick-L2TP-Rx-Connect-Speed (43).

IPoE

IP-over-Ethernet (IPoE) is a popular alternative to PPPoE based access using DHCP for IPv4 and DHCPv6 for IPv6 where both protocols are handled in the IPoE daemon (ipoed).

IPoE subscribers are identified by IFP, optional VLAN and client MAC address.

The dynamic creation of IPoE subscribers is triggered by the first DHCPv4 discovery or DHCPv6 solicit request received. Any response is postponed until the subscriber is successfully authenticated using known authentication methods like none, local or RADIUS. For DHCP mode server all addresses are assigned during authentication to the subscriber and used by DHCPv4 and DHCPv6 to handle client requests. For the DHCP relay mode, all IP addresses are allocated by an external DHCP server.

IPoE subscribers will be terminated automatically if all protocol bindings are deleted.

IPoE 1:1 VLAN Support

RBFS supports both 1:1 VLAN (VLAN Per Subscriber) model for subscriber traffic for IPoE subscribers.

1:1 VLAN (VLAN-per-subscriber) Model

In the 1:1 VLAN model, there is a unique dedicated customer VLAN for a subscriber, that is one VLAN per Subscriber. This model establishes a unique path between each subscriber interface and the router for data transmission by providing traffic separation for every subscriber.

1:1 model operations are relatively simple as it provides one-to-one mapping of specific VLANs to specific subscribers. New services can be added easily without affecting other subscribers and services with this model. However, in a large-scale deployment, this model demands highly scalable and robust routers that can manage many hundreds of VLANs.

The following diagram shows a dedicated customer VLAN for a subscriber, that is, one VLAN per Subscriber.

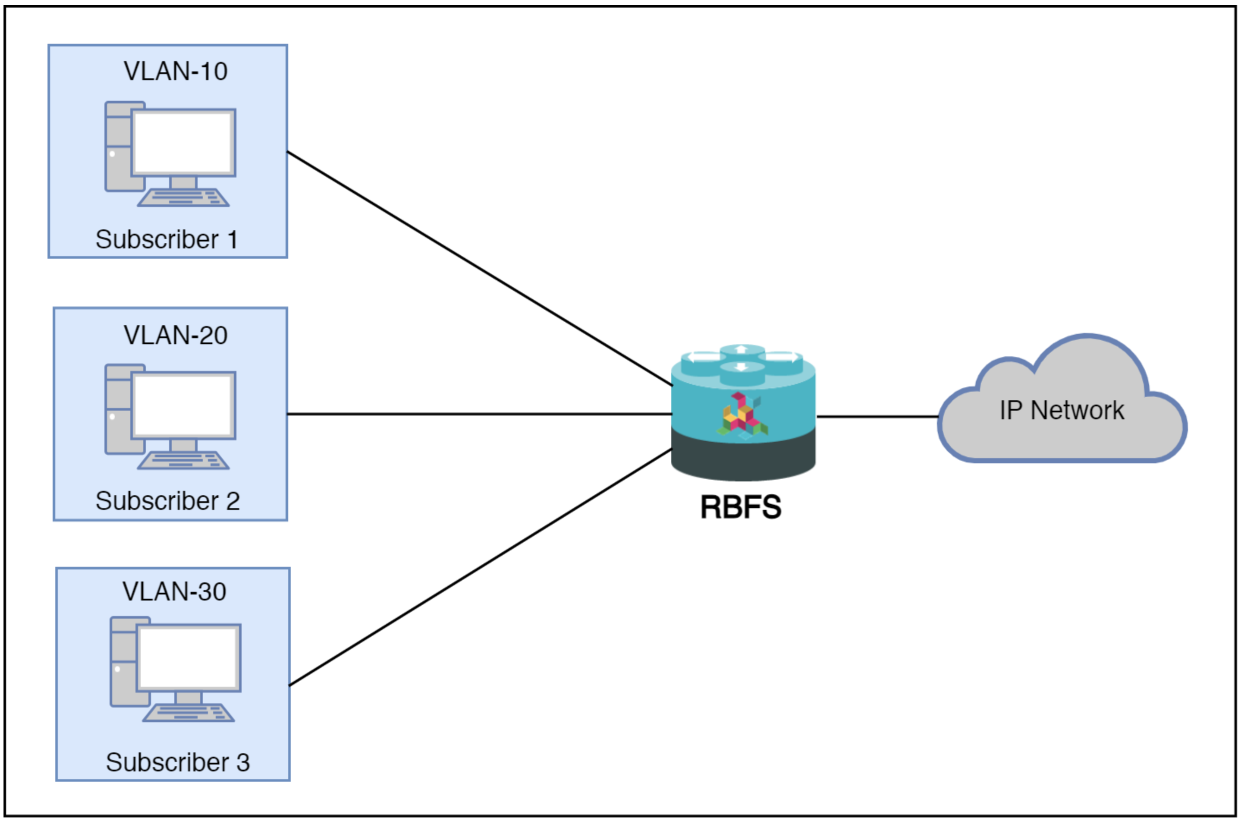


Figure 4. 1:1 VLAN (VLAN-per-subscriber)

N:1 VLAN (Shared VLAN) Model

The Shared VLAN model provides many-to-one (N:1) subscriber-to-service connectivity. This model provides a single VLAN to many subscribers. Unlike in the 1:1 model, in which the VLAN is dedicated to a customer, N:1 provides a shared VLAN to many subscribers, and this VLAN carries all types of service (i.e., data, voice, video, etc.). One disadvantage of shared VLAN is the lack of logical isolation between user sessions at the VLAN level.

The following diagram shows a single VLAN that is connected to many subscribers.

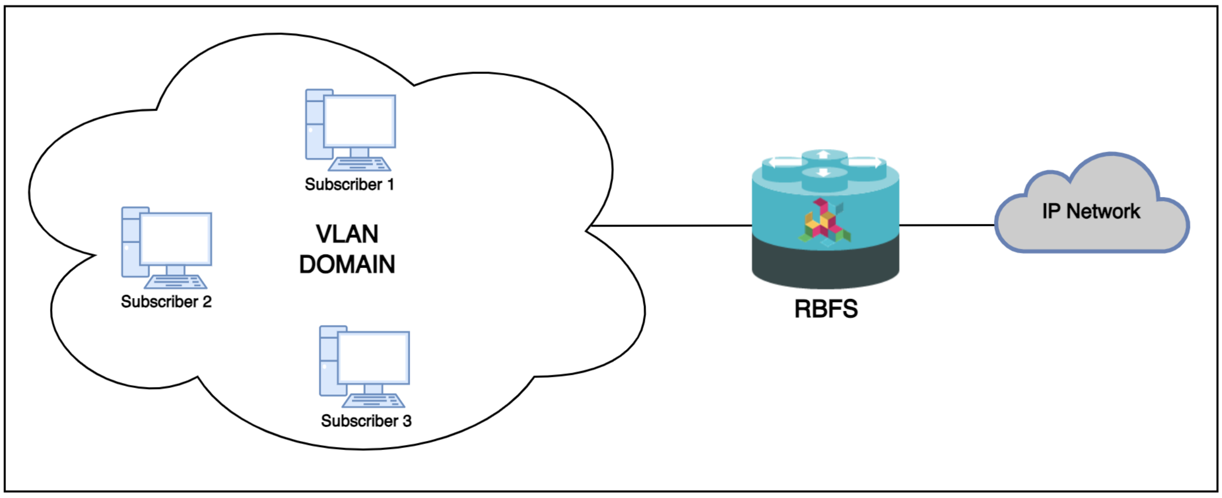


Figure 5. N:1 VLAN (Shared VLAN-per-service)

Limitations for IPoE N:1 VLAN

Access services such as http-redirect and subscriber-acl are not supported for IPoE N:1 subscribers.

IPoE Session Limit

A customer line is typically represented by one (single-tagged) or two VLAN (double-tagged) on a physical interface with a limitation to one subscriber, which is also called the 1:1 VLAN mode.

IPoE subscribers are implicitly limited to max one per MAC address, as client MAC address is used as part of the key to identify subscribers.

IPoE Username

For each IPoE subscriber, a username is generated automatically using the client’s MAC address followed by @ipoe.

Example: fe:08:e8:ea:1d:32@ipoe

IPoE DHCPv6 Server DUID

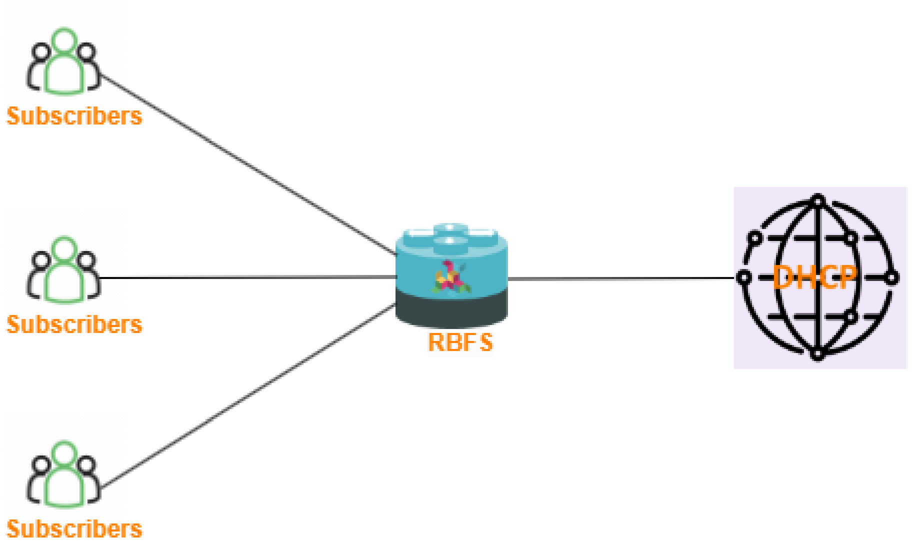
The generated DHCPv6 server identifier DUID is from type 3 (DUID-LL) with hardware type 27 (EUI64) and IFP MAC address to derive the EUI64 interface identifier.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      DUID-Type 3 (DUID-LL)     |    hardware type 27 (EUI64)   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      interface-identifier                      |
|                                                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPoE DHCP Relay

When IPoE is configured on relay mode, the system relies on an external DHCP server for IP address allocation and client configuration functions. The DHCP server contains a pool of IP addresses, and from that pool, it allocates addresses to subscribers. RBFS acts as a DHCP relay during its interaction with the DHCP server. The feature allows multiple RBFS instances to use a single centralized DHCP server for their IP allocation.

For information on how to configure IPoE in relay mode, see the section DHCPv4.



The following table provides an overview of all options supported by RBFS.

IPoE - DHCPv4 Options Supported

In relay mode, unsupported options are still forwarded along with the supported ones.

DHCP Options

Options	Sub-options	Description	Comment
0	-	Pad	-
1	-	Subnet Mask	-
3	-	Router	-
6	-	Domain Server	-
50	-	Address Request	-
51	-	Address Time	-
53	-	DHCP message type	-
54	-	DHCP Server ID	-
55	-	Parameter list	-
56	-	DHCP message	-
61	-	Client ID	-
82	-	Relay agent information	-
82	1	Agent circuit ID	-
82	2	Agent Remote ID	-
82	9	Vendor Specific	-
255	-	End	-

The following table provides vendor-specific relay options (DHCP Option 82, sub-option 9). Access line attributes within Broadband Forum (3561) options are recognized by RBFS as access line attributes and are also forwarded to RADIUS.

DHCP Vendor Options

Vendor Options	Sub-options	Description	Comment
3561	129	Actual Data Rate Upstream	-

Vendor Options	Sub-options	Description	Comment
3561	130	Actual Data Rate Downstream	-
3561	131	Minimum Data Rate Upstream	-
3561	132	Minimum Data Rate Downstream	-
3561	133	Attainable DataRate Upstream	-
3561	134	Attainable DataRate Downstream	-
3561	135	Maximum Data Rate Upstream	-
3561	136	Maximum Data Rate Downstream	-
3561	137	Minimum Data Rate Upstream in low power state	-
3561	138	Minimum Data Rate Downstream in low power state	-
3561	139	Maximum Interleaving Delay Upstream	-
3561	140	Actual Interleaving Delay Upstream	-
3561	141	Maximum Interleaving Delay Downstream	-
3561	142	Actual Interleaving Delay Downstream	-
3561	144	Data Link Encapsulation	-
3561	145	DSL Type	-
3561	146	PON-Access-Type	Backward compatibility with LIHAWI draft version 00.
3561	147	ONT/ONU-Average-Data-Rate-Downstream	Backward compatibility with LIHAWI draft version 00.
3561	148	ONT/ONU-Peak-Data-Rate-Downstream	Backward compatibility with LIHAWI draft version 00.

Vendor Options	Sub-options	Description	Comment
3561	149	ONT/ONU-Maximum-Data-Rate-Upstream	Backward compatibility with LIHAWI draft version 00.
3561	150	ONT/ONU-Assured-Data-Rate-Upstream	Backward compatibility with LIHAWI draft version 00.
3561	151	PON-Access-Type / PON-Tree-Maximum-Data-Rate-Upstream	If value is > 999, RBFS interprets as PON-Tree-Maximum-Data-Rate-Upstream as defined by LIHAWI version 00, otherwise as PON-Access-Type.
3561	152	PON-Tree-Maximum-Data-Rate-Downstream	Backward compatibility with LIHAWI draft version 00.
3561	155	Expected Throughput (ETR) Upstream	-
3561	156	Expected Throughput (ETR) Downstream	-
3561	157	Attainable Expected Throughput (ATTETR) Upstream	-
3561	158	Attainable Expected Throughput (ATTETR) Downstream	-
3561	159	Gamma Data Rate (GDR) Upstream	-
3561	160	Gamma Data Rate (GDR) Downstream	-
3561	161	Attainable Gamma Data Rate (ATTGDR) Upstream	-
3561	162	Attainable Gamma Data Rate (ATTGDR) Downstream	-

Vendor Options	Sub-options	Description	Comment
3561	176	ONT/ONU-Average-Data-Rate-Downstream	-
3561	177	ONT/ONU-Peak-Data-Rate-Downstream	-
3561	178	ONT/ONU-Maximum-Data-Rate-Upstream	-
3561	179	ONT/ONU-Assured-Data-Rate-Upstream	-
3561	180	PON-Tree-Maximum-Data-Rate-Upstream	-
3561	181	PON-Tree-Maximum-Data-Rate-Downstream	-

IPoE - DHCPv6 Options Supported

The following table provides the list of all the IPoE DHCPv6 options supported by RBFS. In relay mode, unsupported options are still forwarded along with the supported ones.

DHCP Options

Options	Sub-options	Description	Comment
1	-	OPTION_CLIENTID	-
2	-	OPTION_SERVERID	-
3	-	OPTION_IA_NA	-
6	-	OPTION_ORO	-
8	-	OPTION_ELAPSED_TIME	-
9	-	OPTION_RELAY_MSG	-
13	-	OPTION_STATUS_CODE	-
14	-	OPTION_RAPID_COMMIT	-
17	-	OPTION_VENDOR_OPTS	-

Options	Sub-options	Description	Comment
18	-	OPTION_INTERFACE_ID	-
23	-	OPTION_DNS_SERVERS	-
25	-	OPTION_IA_PD	-
26	1	OPTION_IAPREFIX	-
37	2	OPTION_REMOTE_ID	-

The list below outlines vendor-specific options (DHCP Option 17). Access line attributes within Broadband Forum (3561) options are recognized by RBFS as access line attributes and are also forwarded to RADIUS.

DHCPv6 Vendor Options

Vendor Options	Sub-options	Description	Comment
3561	129	Actual Data Rate Upstream	-
3561	130	Actual Data Rate Downstream	-
3561	131	Minimum Data Rate Upstream	-
3561	132	Minimum Data Rate Downstream	-
3561	133	Attainable DataRate Upstream	-
3561	134	Attainable DataRate Downstream	-
3561	135	Maximum Data Rate Upstream	-
3561	136	Maximum Data Rate Downstream	-
3561	137	Minimum Data Rate Upstream in low power state	-
3561	138	Minimum Data Rate Downstream in low power state	-
3561	139	Maximum Interleaving Delay Upstream	-
3561	140	Actual Interleaving Delay Upstream	-
3561	141	Maximum Interleaving Delay Downstream	-

Vendor Options	Sub-options	Description	Comment
3561	142	Actual Interleaving Delay Downstream	-
3561	144	Data Link Encapsulation	-
3561	145	DSL Type	-
3561	146	PON-Access-Type	Backward compatibility with LIHAWI draft version 00.
3561	147	ONT/ONU-Average-Data-Rate-Downstream	Backward compatibility with LIHAWI draft version 00.
3561	148	ONT/ONU-Peak-Data-Rate-Downstream	Backward compatibility with LIHAWI draft version 00.
3561	149	ONT/ONU-Maximum-Data-Rate-Upstream	Backward compatibility with LIHAWI draft version 00.
3561	150	ONT/ONU-Assured-Data-Rate-Upstream	Backward compatibility with LIHAWI draft version 00.
3561	151	PON-Access-Type / PON-Tree-Maximum-Data-Rate-Upstream	If value is > 999, RBFS interprets as PON-Tree-Maximum-Data-Rate-Upstream as defined by LIHAWI version 00, otherwise as PON-Access-Type.
3561	152	PON-Tree-Maximum-Data-Rate-Downstream	Backward compatibility with LIHAWI draft version 00.
3561	155	Expected Throughput (ETR) Upstream	-

Vendor Options	Sub-options	Description	Comment
3561	156	Expected Throughput (ETR) Downstream	-
3561	157	Attainable Expected Throughput (ATTETR) Upstream	-
3561	158	Attainable Expected Throughput (ATTETR) Downstream	-
3561	159	Gamma Data Rate (GDR) Upstream	-
3561	160	Gamma Data Rate (GDR) Downstream	-
3561	161	Attainable Gamma Data Rate (ATTGDR) Upstream	-
3561	162	Attainable Gamma Data Rate (ATTGDR) Downstream	-
3561	176	ONT/ONU-Average-Data-Rate-Downstream	-
3561	177	ONT/ONU-Peak-Data-Rate-Downstream	-
3561	178	ONT/ONU-Maximum-Data-Rate-Upstream	-
3561	179	ONT/ONU-Assured-Data-Rate-Upstream	-
3561	180	PON-Tree-Maximum-Data-Rate-Upstream	-
3561	181	PON-Tree-Maximum-Data-Rate-Downstream	-

5.1.2. Subscriber Management Configuration Overview

RBFS Subscriber Management configuration involves setting up various profiles and parameters that control how subscribers interact with network services. This includes authentication, service access, protocol handling, and other functions organized under a structured configuration hierarchy. The profiles determine how subscribers authenticate, access network services, and use various protocols. The configuration starts with mandatory settings such as interface, access, and AAA profiles, followed by optional configurations for more features such as RADIUS, L2TP, and service profiles. Each profile plays a crucial role in ensuring efficient and secure subscriber management.

Configuration Hierarchy

The configuration of physical interfaces (IFP) and their associated VLANs is managed through a set of profiles that define parameters for various functions. These include authentication settings with AAA, service management for protocols such as IGMP and MLD, and access protocols such as PPPoE.

The following image illustrates how the subscriber management configuration and profile system are organized.

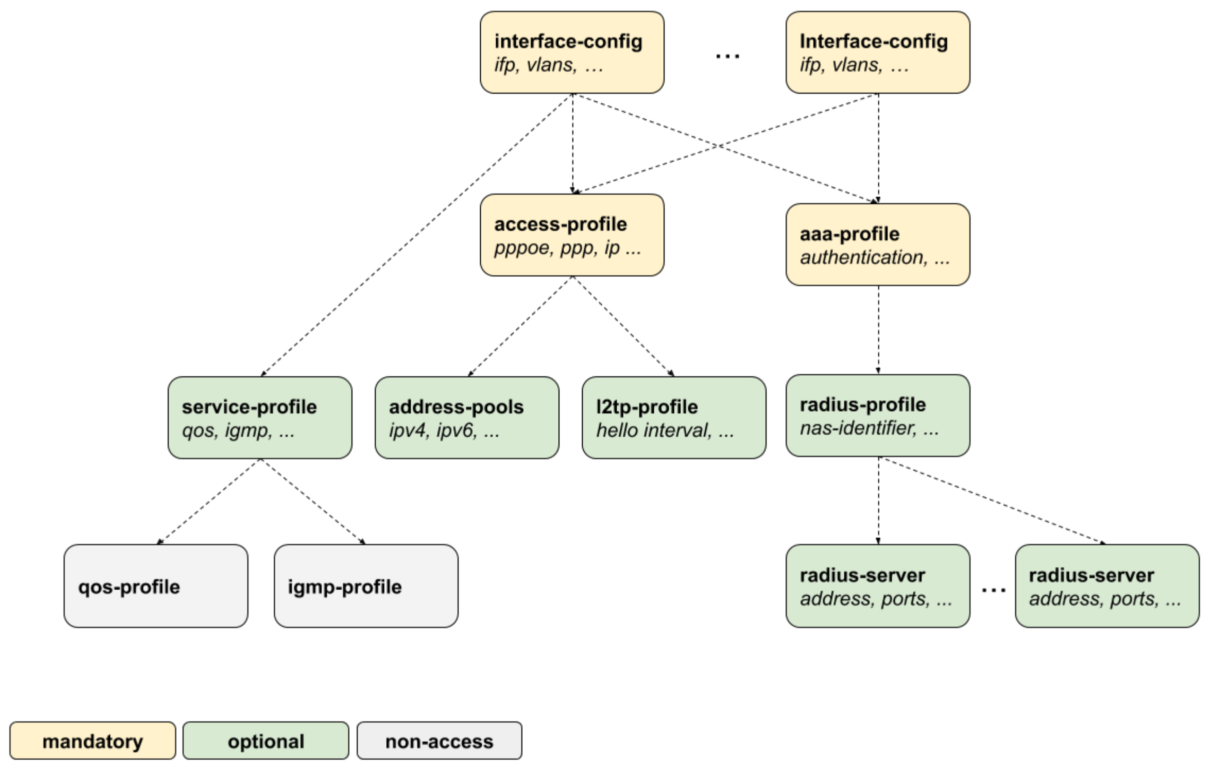


Figure 6. Configuration and Profiles

All subscriber management configurations and profiles are managed under the top-level hierarchy access. This hierarchy acts as the central point for defining and managing various access protocols, subscriber management profiles, and authentication settings that are crucial for network operations.

The following is the access command and all the high-level options available for Subscriber Management configurations.

```
supervisor@switch: cfg> set access
  <cr>
  aaa-profile           Global AAA profile configuration
  access-profile        Global access profile configuration
  chassis-id            Chassis ID for this node [Range: <0-15>]
  dhcp-relay            Global DHCP relay configuration
  dhcp-server           Global DHCP server configuration
  dhcpv6-server         Global DHCP server configuration
  interface             Global interface profile configuration
  l2bsa                 Global access l2bsa configurations
  l2tp-pool             Global L2TPv2 pool configuration
  l2tp-profile          Global L2TPv2 profile configuration
  pool                  Global address pool configuration
  radius-profile        Global AAA RADIUS profile configuration
  radius-server         Global RADIUS server configuration
  service-profile       Global service profile configuration
  user-profile          Global user profile configuration
```

In the following sections, you will find comprehensive descriptions and steps for each configuration. The process begins with setting up the access interface configuration, which is the first step. This is then followed by other important access profile and AAA profile configurations, which are crucial for managing subscriber access.

- **interface-config** Access Interface Configuration
- **access-profile** Access Profile Configuration
- **aaa-profile** AAA Profile Configuration

The second part covers the optional configurations:

- **radius-profile** RADIUS Profile Configuration
- **radius-server** RADIUS Server Configuration
- **service-profile** Service Profile Configuration
- **l2tp-profile** L2TP Profile Configuration
- **address-pools** Address Pool Configuration

The user-profile and l2tp-pool are the only components not referenced by name. The key here is the user or pool name.

- **user-profile** User Profile Configuration

- **l2tp-pool** L2TP Tunnel Pool Configuration

Access Interface Configuration

Table: global.access.interface.config

While there is no single specific way to configure subscriber management, it is ideal to start with mandatory configurations and then move on to optional ones. The access interface configuration is the anchor point for almost all further access configurations. The interface configuration defines the access type and access profile. For more information, see sections Access Profile Configuration, AAA profile AAA Profile Configuration.

Multiple interface configurations per IFP with disjoint VLAN ranges are supported.

The following image illustrates the access interface configurations and how they are associated with the entire subscriber management.

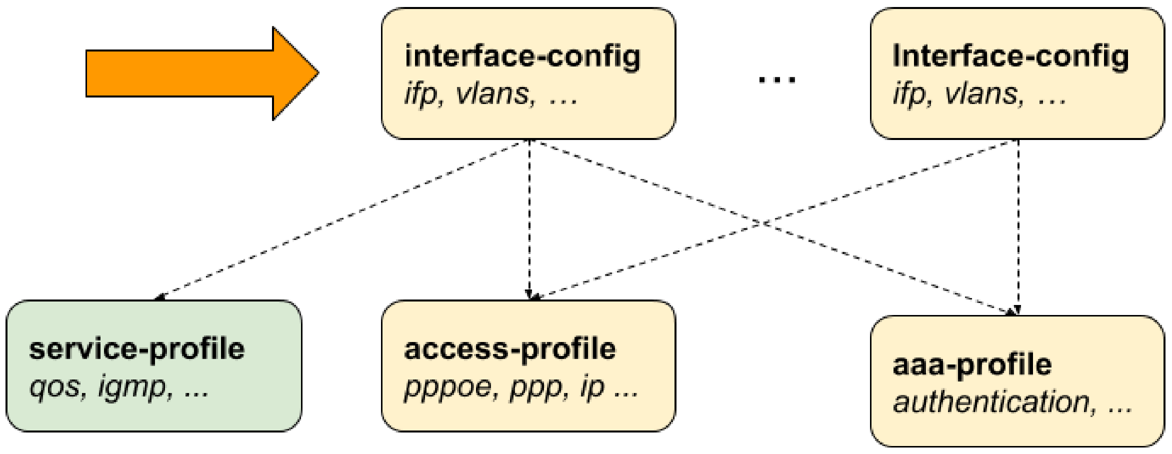


Figure 7. Access Interface Configuration

You can configure multiple interfaces for access and subscriber management, and each interface can reference the same profiles. This allows for efficient and scalable network management.

To configure the access interface, you must complete the following major tasks.

1. Configure the physical interface name (IFP or LAG) and VLAN range
2. Configure the mandatory access type (PPPoE or IPoE)
3. Configure the mandatory access profile

4. Configure the mandatory AAA profile
5. Configure optional attributes such as service profile or session limit

Configuring Access Interfaces

Access interfaces can be configured without VLAN tags (untagged VLAN) and with one VLAN tag (single-tagged) or with two VLAN tags (double-tagged).

The following are the command and command options available to configure Access Interface.

```
supervisor@switch: cfg> set access interface
  <cr>
  double-tagged         Double tagged access
  single-tagged         Single tagged access
  untagged              Untagged access

supervisor@switch: cfg> set access interface untagged ifp-0/0/0
  <cr>
  aaa-profile-name          AAA profile name
  access-profile-name       Access profile name
  access-type               Access service type
  dhcp-min-elapsed-time     DHCPv4/v6 minimum elapsed time in seconds [Range: <1-
600>]
  gateway-ifl               IPoE gateway IFL (unnumbered source IFL)
  max-subscribers-per-mac   Restrict maximum subscribers per MAC address [Range:
<1-65535>]
  max-subscribers-per-vlan  Restrict maximum subscribers per VLAN [Range: <1-
65535>]
  pppoe-pado-delay          PPPoE PADO delay in seconds [Range: <1-255>]
  redundancy-session-id     Redundancy session ID for this interface [Range: <1-
65535>]
  service-profile-name      Service profile name
  vlan-profile-enable       Enable VLAN profiles
```

Example:

In the following example configuration, the untagged access interface 'ifp-0/0/0' is configured for PPPoE access type, with the Access Profile set to 'pppoe-dual', Service Profile set to 'service-profile1', and AAA Profile set to 'aaa-radius'. The option 'vlan-profile-enable' is enabled by setting it to 'true', and the parameters 'max-subscribers-per-vlan' and 'max-subscribers-per-mac' are both assigned the value of '1'.

```
supervisor@switch: cfg> show config access interface untagged ifp-0/0/0
{
  "rtbrick-config:untagged": {
    "interface-name": "ifp-0/0/0",
    "access-type": "PPPoE",
    "access-profile-name": "pppoe-dual",
    "service-profile-name": "service-profile1",
    "aaa-profile-name": "aaa-radius",
    "vlan-profile-enable": "true",
    "max-subscribers-per-vlan": 1,
    "max-subscribers-per-mac": 1
  }
}
```

Attribute	Description
access-type	Defines the access protocol used for this interface. This is a mandatory attribute. **Values:** PPPoE or IPoE
access-profile-name	Specifies the name of the access profile (mandatory). For more information, see Access Profile Configuration.
aaa-profile-name	Specifies the name of the AAA profile (mandatory). For more information, see AAA Profile Configuration.
service-profile-name	This option allows assigning an optional service profile which can be dynamically overwritten via RADIUS. For more information, see Service Profile Configuration .
max-subscribers-per-vlan	This option defines the maximum number of subscribers per IFP and VLAN. A value of '1' will implicitly set the VLAN mode to 1:1 VLAN mode, where any value greater than 1 indicates N:1 VLAN mode. **Default:** 1 **Range:** 1 - 65535
max-subscribers-per-mac	Maximum number of subscribers per IFP, VLAN, and MAC address. This option must be less or equal specified for the 'max-subscribers-per-vlan' attribute. This option does not affect IPoE subscribers, as they are inherently limited to a maximum of one per MAC address. **Default:** 1 **Range:** 1 - 65535

Attribute	Description
outer-vlan-encapsulation	Limits the outer VLAN encapsulation to 802.1ad (TPID 0x88a8) or 802.1q (TPID 0x8100). By default, RBFS uses an autosensing mechanism that determines the correct TPID based on the first subscriber-initiated packet, such as PPPoE Discovery (PADI, PADR), DHCP discovery, or DHCPv6 solicit. **Default:** autosensing **Values:** 802.1ad or 802.1q
vlan-profile-enable	If enabled, incoming PPPoE sessions (PPPoE PADI/PADR) are not honored unless matching vlan-profile is found in the table global.vlan.profile of the PPPoE daemon. For more information about VLAN profiles, see PPPoE VLAN Profiles. **Default:** false
gateway-ifl	This options selects the IPoE gateway IFL (unnumbered source IFL) which is typically a loopback interface used as a gateway for IPoE subscribers.
pppoe-pado-delay	Specifies the PPPoE PADO delay, in seconds. This setting allows you to specify a wait time in seconds after receiving a PPPoE Active Discovery Initiation (PADI) control packet from a PPPoE client before sending a PPPoE Active Discovery Offer (PADO) packet to indicate that it can serve the client request. **Default:** disabled **Range:** 1 - 255 The PPPoE PADO delay is used for stateless high availability.
dhcp-min-elapsed-time	Defines the minimum elapsed time, in seconds, for DHCPv4/DHCPv6 transactions. This setting ensures that a DHCP request must meet the specified minimum duration before it is processed. **Default:** disabled **Range:** 1 - 600 The DHCPv4/v6 minimum elapsed time is used for stateless high availability.

Attribute	Description
redundancy-session-id	Specifies the redundancy session ID required for stateful high availability. **Default:** disabled **Range:** 1 - 65535

Configuring Untagged Interfaces

The following command and options are used to configure the untagged access interface.

```
supervisor@switch: cfg> set access interface untagged
  <interface-name>      Name of the physical interface

supervisor@switch: cfg> set access interface untagged ifp-0/0/0
  <cr>
  aaa-profile-name          AAA profile name
  access-profile-name       Access profile name
  access-type               Access service type
  dhcp-min-elapsed-time     DHCPv4/v6 minimum elapsed time in seconds [Range: <1-
600>]
  gateway-ifl               IPoE gateway IFL (unnumbered source IFL)
  max-subscribers-per-mac   Restrict maximum subscribers per MAC address [Range:
<1-65535>]
  max-subscribers-per-vlan  Restrict maximum subscribers per VLAN [Range: <1-
65535>]
  pppoe-pado-delay          PPPoE PADO delay in seconds [Range: <1-255>]
  redundancy-session-id     Redundancy session ID for this interface [Range: <1-
65535>]
  service-profile-name      Service profile name
  vlan-profile-enable       Enable VLAN profiles

supervisor@switch: cfg> set access interface untagged ifp-0/0/0 access-type PPPoE
supervisor@switch: cfg> set access interface untagged ifp-0/0/0 access-profile-
name pppoe-dual
supervisor@switch: cfg> set access interface untagged ifp-0/0/0 aaa-profile-name
aaa-radius
supervisor@switch: cfg> commit
supervisor@switch: cfg> show config access interface untagged ifp-0/0/0
{
  "rtbrick-config:untagged": {
    "interface-name": "ifp-0/0/0",
    "access-type": "PPPoE",
    "access-profile-name": "pppoe-dual",
    "aaa-profile-name": "aaa-radius"
  }
}
```

- Untagged interfaces are not supported on Broadcom QAX platforms.
- The access interface-name can be a physical IFP or link

aggregation (LAG).

Configuring Single VLAN Tagged Interfaces

You need to define the VLAN identifier range within the range from 128 to 4000 for VLAN tagged interface. The following command and options are used to configure a single VLAN-tagged interface.

```
supervisor@switch: cfg> set access interface single-tagged
  <interface-name>      Name of the physical interface

supervisor@switch: cfg> set access interface single-tagged ifp-0/0/0
  <outer-vlan-min>      Outer VLAN min [Range: <1-4094>]

supervisor@switch: cfg> set access interface single-tagged ifp-0/0/0 128
  <outer-vlan-max>      Outer VLAN max [Range: <1-4094>]

supervisor@switch: cfg> set access interface single-tagged ifp-0/0/0 128 3000
  <cr>
  aaa-profile-name          AAA profile name
  access-profile-name       Access profile name
  access-type               Access service type
  dhcp-min-elapsed-time     DHCPv4/v6 minimum elapsed time in seconds [Range: <1-
600>]
  gateway-ifl               IPoE gateway IFL (unnumbered source IFL)
  max-subscribers-per-mac   Restrict maximum subscribers per MAC address [Range:
<1-65535>]
  max-subscribers-per-vlan  Restrict maximum subscribers per VLAN [Range: <1-
65535>]
  outer-vlan-encapsulation  Outer VLAN encapsulation
  pppoe-pado-delay          PPPoE PADO delay in seconds [Range: <1-255>]
  redundancy-session-id     Redundancy session ID for this interface [Range: <1-
65535>]
  service-profile-name      Service profile name
  vlan-profile-enable       Enable VLAN profiles
```

In the following example, the single tagged access interface ifp-0/0/0 is configured with the Outer VLAN minimum value 128 and the outer VLAN maximum value 3000. The Access Type is defined PPPoE, Access Profile pppoe-dual, and AAA Profile as aaa-radius.

```
supervisor@switch: cfg> set access interface single-tagged ifp-0/0/0 128 3000
access-type PPPoE
supervisor@switch: cfg> set access interface single-tagged ifp-0/0/0 128 3000
access-profile-name pppoe-dual
supervisor@switch: cfg> set access interface single-tagged ifp-0/0/0 128 3000 aaa-
profile-name aaa-radius
supervisor@switch: cfg> commit
supervisor@switch: cfg> show config access interface single-tagged ifp-0/0/0 128
3000
{
  "rtbrick-config:single-tagged": [
    {
      "interface-name": "ifp-0/0/0",
      "outer-vlan-min": 128,
      "outer-vlan-max": 3000,
      "access-type": "PPPoE",
      "access-profile-name": "pppoe-dual",
      "aaa-profile-name": "aaa-radius"
    }
  ]
}
```

The access interface-name can be a physical IFP or LAG.

Configuring Double Tagged VLAN Interfaces

When configuring double-tagged VLAN interfaces, setting the minimum and maximum VLAN ID to the same value ensures that only a specific VLAN ID is matched. It indicates that the VLAN interface will specifically recognize and process traffic tagged with that exact VLAN ID.

The following commands and options are used to configure double-tagged VLAN interfaces.

```
supervisor@switch: cfg> set access interface double-tagged
  <interface-name>      Name of the physical interface

supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0
  <outer-vlan-min>      Outer VLAN min [Range: <1-4094>]

supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128
  <outer-vlan-max>      Outer VLAN max [Range: <1-4094>]

supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128 3000
  <inner-vlan-min>      Inner VLAN min [Range: <1-4094>]

supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128 3000 7
  <inner-vlan-max>      Inner VLAN max [Range: <1-4094>]

supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128 3000 7 7
  <cr>
  aaa-profile-name          AAA profile name
  access-profile-name       Access profile name
  access-type               Access service type
  dhcp-min-elapsed-time     DHCPv4/v6 minimum elapsed time in seconds [Range: <1-
600>]
  gateway-ifl               IPoE gateway IFL (unnumbered source IFL)
  max-subscribers-per-mac   Restrict maximum subscribers per MAC address [Range:
<1-65535>]
  max-subscribers-per-vlan  Restrict maximum subscribers per VLAN [Range: <1-
65535>]
  outer-vlan-encapsulation  Outer VLAN encapsulation
  pppoe-pado-delay          PPPoE PADO delay in seconds [Range: <1-255>]
  redundancy-session-id     Redundancy session ID for this interface [Range: <1-
65535>]
  service-profile-name      Service profile name
  vlan-profile-enable        Enable VLAN profiles
```

In the following example, the double-tagged access interface ifp-0/0/0 is configured with the Outer VLAN minimum value 128 and the outer VLAN maximum value 3000. The configuration also defines the inner VLAN minimum value 7 and Inner VLAN maximum value 7. The Access Type is defined PPPoE, Access Profile pppoe-dual, and AAA Profile aaa-radius.

```
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128 3000 7 7
access-type PPPoE
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128 3000 7 7
access-profile-name pppoe-dual
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 128 3000 7 7
aaa-profile-name aaa-radius
supervisor@switch: cfg> commit
supervisor@switch: cfg> show config access interface single-tagged ifp-0/0/0 128
3000 7 7
{
  "rtbrick-config:double-tagged": {
    "interface-name": "ifp-0/0/0",
    "outer-vlan-min": 128,
    "outer-vlan-max": 3000,
    "inner-vlan-min": 7,
    "inner-vlan-max": 7,
    "access-type": "PPPoE",
    "access-profile-name": "pppoe-dual",
    "aaa-profile-name": "aaa-radius"
  }
}
```

The access interface-name can be a physical IFP or LAG.

The following example sets a PPPoE PADO delay of 30 seconds for a double-tagged interface ifp-0/0/0.

```
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 201 250 201
250
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 201 250 201
250 access-type PPPoE
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 201 250 201
250 access-profile-name pppoe-default-ds
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 201 250 201
250 aaa-profile-name aaa-profile
supervisor@switch: cfg> set access interface double-tagged ifp-0/0/0 201 250 201
250 pppoe-pado-delay 30
```

Access Profile Configuration

You must configure an access interface with an access profile name, such as pppoe-dual and it is also essential to configure the properties and parameters of the access profile itself.

The following image illustrates the access interface configurations and how they are associated with the entire subscriber management configurations.

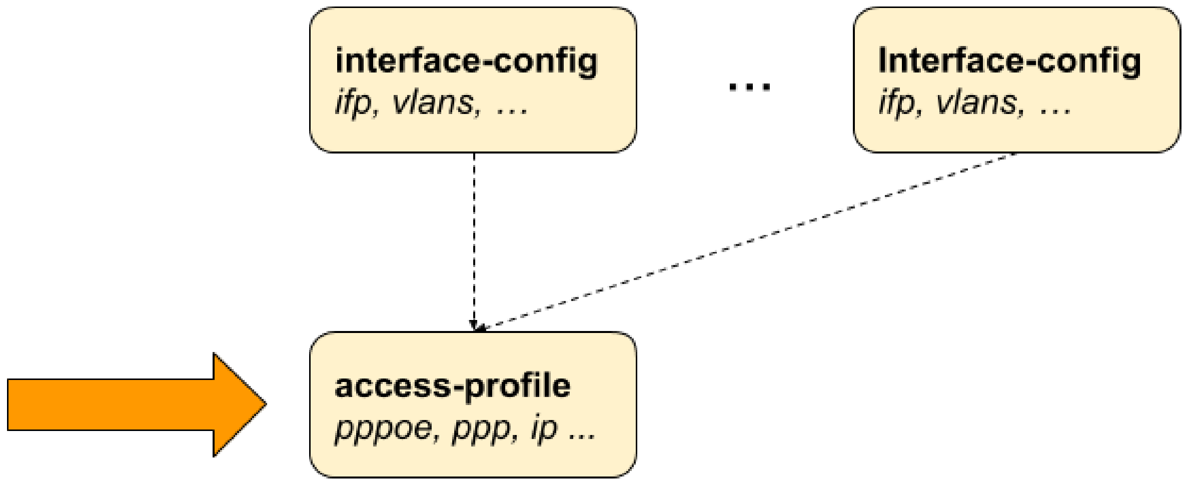


Figure 8. Access Profile Configuration

Configuring the Access Profile

Configuring an access profile involves specifying its name and defining various parameters to control how it handles network traffic and client interactions. The following command and options allow you to configure an access profile.

```
supervisor@switch: cfg> set access access-profile
  <profile-name>         Name of the access profile

supervisor@switch: cfg> set access access-profile pppoe-dual
  <cr>
  address-family         Address-family configuration
  instance               Instance name
  protocol               Protocol configuration
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
instance	Change routing instance. **Default:** default

PPPoE with IPv4 and IPv6

The following access profile configuration is for a PPPoE access profile named pppoe-dual that supports both IPv4 and IPv6. The instance is marked as default, and the session protection is enabled for PPPoE by setting the value to true, and the VLAN priority is set to 6.

The configuration defines various PPP parameters such as LCP, IPCP, and so on. For LCP (Link Control Protocol), the authentication protocol is specified as PAP_CHAP, indicating that both PAP (Password Authentication Protocol) and CHAP (Challenge Handshake Authentication Protocol) are allowed. The echo-interval, which is the interval in seconds for sending LCP echo requests to check the connection, is specified as 30 seconds. The echo-max-retransmit parameter, which is the maximum number of retransmissions for LCP echo requests before considering the link down, is set to 3. The configuration enables the LCP echo requests by specifying true for the echo-enable parameter.

In the configuration, both the IPCP and IP6CP are enabled by setting their values to true. These protocols are used to configure both the IPv4 and IPv6 settings over PPP. The source IFL (logical interface) for IPCP is specified as 'lo-0/0/0/1'. The RA (Router Advertisement) Configuration is enabled by setting the value to 'true', which allows the router to send RA messages for IPv6 configuration. The interval period for the RA messages is set to 60 seconds. DHCPv6 is enabled by setting the value to true, which allows the device to assign IPv6 addresses to the subscribers through DHCP.

The L2TP tunnel profile is defined as l2tp-default, which allows the encapsulation of PPP sessions over IP networks. Both IPv4 and IPv6 address families are enabled by setting their values to true using the address family configurations. For IPv4, the primary DNS server is specified '198.51.100.1' and the secondary DNS server is specified '198.51.100.4'. For IPv6, the primary DNS server is specified as '2001:db8:0:100::'' and the secondary DNS server is specified '2001:db8:0:104::'.

```
supervisor@switch: cfg> show config access access-profile pppoe-dual
{
  "rtbrick-config:access-profile": {
    "profile-name": "pppoe-dual",
    "instance": "default",
    "protocol": {
      "pppoe": {
        "enable": "true",
        "session-protection": {
          "enable": "true"
        },
        "vlan-priority": 6
      },
      "ppp": {
        "lcp": {
          "authentication-protocol": "PAP_CHAP",
          "echo-interval": 30,
          "echo-max-retransmit": 3,
          "echo-enable": "true"
        },
        "ipcp": {
          "enable": "true",
          "source-ifl": "lo-0/0/0/1"
        },
        "ip6cp": {
          "enable": "true"
        }
      },
      "ra": {
        "enable": "true",
        "interval": 60
      },
      "dhcpv6": {
        "enable": "true"
      },
      "l2tp": {
        "tunnel-profile": "l2tp-default"
      }
    },
    "address-family": {
      "ipv4": {
        "enable": "true",
        "primary-dns": "198.51.100.1",
        "secondary-dns": "198.51.100.4"
      },
      "ipv6": {
        "enable": "true",
        "primary-dns": "2001:db8:0:100::",
        "secondary-dns": "2001:db8:0:104::"
      }
    }
  }
}
```

IPoE with IPv4 and IPv6

The example configuration below sets up an IPoE access profile named 'ipoe-dual' to handle IPoE sessions, which support both IPv4 and IPv6 address families. The router acts as a DHCP server for both protocols, assigning IP addresses and DNS server information to client devices.

The profile name is ipoe-dual. By setting the value to true, this profile enables both DHCP and DHCPv6 protocols. The mode is defined as server for both DHCP and DHCPv6, which allows the router to act as a server for IPv4 and IPv6. The IPv4 address family is configured for the access profile by setting the value to true. For IPv4, proxy ARP is enabled by setting proxy-arp-enable to true.

The IPv4 address family is configured for the access profile by setting the value to ‘true. For IPv4, proxy ARP is enabled by setting proxy-arp-enable to true. This allows the router to respond to ARP requests on behalf of other hosts.

The pool name ipoe indicates the IPv4 address allocation to client devices. The primary DNS server for IPv4 is set to '198.51.100.1', and the secondary DNS server is specified as '198.51.100.4'.

The configuration includes various parameters for IPv6. The IPv6 pool name is ipoe-ia-na, which provides IPv6 addresses to subscribers. The prefix-delegation-pool-name parameter is set as ipoe-ia-pd for IPv6 prefix delegation. The primary and secondary DNS servers for IPv6 are specified as '2001:db8:0:100::' and '2001:db8:0:104::' respectively.

```
supervisor@switch: cfg> show config access access-profile ipoe-dual
{
  "rtbrick-config:access-profile":{
    "profile-name":"ipoe",
    "protocol":{
      "dhcp":{
        "enable":"true",
        "mode":"server"
      },
      "dhcpv6":{
        "enable":"true",
        "mode":"server"
      }
    },
    "address-family":{
      "ipv4":{
        "enable":"true",
        "proxy-arp-enable": "true",
        "pool-name":"ipoe",
        "primary-dns":"198.51.100.1,
        "secondary-dns":"198.51.100.4"
      },
      "ipv6":{
        "enable":"true",
        "pool-name":"ipoe-ia-na",
        "prefix-delegation-pool-name":"ipoe-ia-pd",
        "primary-dns": "2001:db8:0:100::",
        "secondary-dns": "2001:db8:0:104::"
      }
    }
  }
}
```

Unrestricted Proxy-ARP Support for Subscriber IFLs

A configuration option is available for subscriber IFL to enable the unrestricted proxy-ARP to reply to all ARP requests from subscribers with the MAC address of the corresponding subscriber IFL.

Syntax:

set access access-profile <access-profile1> **address-family ipv4 proxy-arp** [any | disabled | subscriber]

Sample command:

```
set access access-profile access-profile1 address-family ipv4 proxy-arp any
```

Example:

```
supervisor@rtbrick: cfg> show config access access-profile access-profile1
{
  "rtbrick-config:access-profile": [
    {
      "profile-name": "access-profile1",
      "protocol": {
        "ra": {
          "enable": "true"
        },
        "dhcp": {
          "enable": "true",
          "mode": "server"
        },
        "dhcpv6": {
          "enable": "true",
          "mode": "server"
        }
      },
      "address-family": {
        "ipv4": {
          "enable": "true",
          "pool-name": "pool1",
          "proxy-arp": "any"
        },
        "ipv6": {
          "enable": "true",
          "pool-name": "pool1",
          "prefix-delegation-pool-name": "pool2",
          "primary-dns": "1::1"
        }
      }
    }
  ]
}
```

Configuring IPv4

To make IPv4 available for access protocols such as PPP (PPPoE) or DHCP (IPoE), you must explicitly enable the IPv4 address family within the access profile.

```
supervisor@switch: cfg> set access access-profile pppoe-dual address-family ipv4
  <cr>
  enable                      Enable IPv4
  pool-name                   Local IPv4 pool name
  primary-dns                 Primary DNS server
  proxy-arp-enable            Enable Proxy ARP
  secondary-dns               Secondary DNS server
  static-ipv4                 Static address
  dad-enable                  Enable/disable IPv4 duplicate address detection (enabled
by default)
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable IPv4 **Default:** false
pool-name	The pool-name option (optional) allows assigning the IPv4 address from a locally managed pool. For more details, see Address Pool Configuration. This address is used by protocols like PPP IPCP (PPPoE) or DHCP (IPoE) as a client or peer IPv4 address.
primary-dns secondary-dns	The primary DNS and secondary DNS servers configured are used by protocols such as PPP (PPPoE) or DHCP (IPoE) and advertised to the client.
proxy-arp-enable	Enable (or disable) proxy ARP support for IPoE subscribers. When proxy ARP is enabled, if the RBFS device receives an ARP request from the subscriber for which it has a route to the target (destination) IP address, the RBFS device responds by sending a proxy ARP reply packet containing its own MAC address. The host/subscriber that sent the ARP request then sends the actual destined packets to RBFS, which forwards them to the intended destination. **Default**: NONE.

Attribute	Description
static-ipv4	The static-ipv4 attribute is used to configure a fixed IPv4 address for all clients connected under a specific access profile. This is particularly useful in scenarios requiring a consistent IP address for services. ⚠ This feature should be only used with caution.
dad-enable	With the option, you can enable or disable the Duplicate Address Detection (DAD) for IPv4 addresses. It prevents address conflicts by verifying that an IPv4 address is not already in use on the network before it is assigned to an interface. **Default:** true

Configuring IPv6

To make IPv6 available for access protocols such as PPP (PPPoE) or DHCP (IPoE), you must explicitly enable the IPv4 address family within the access profile.

```
supervisor@switch: cfg> set access access-profile pppoe-dual address-family ipv6
  <cr>
  enable                         Enable IPv6
  pool-name                      Local IPv6 pool name
  prefix-delegation-pool-name    Local IPv6 prefix delegation pool name
  primary-dns                    Primary DNS server
  secondary-dns                  Secondary DNS server
  dad-enable                     Enable/disable IPv6 duplicate address detection
(enabled by default)
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable IPv6 **Default:** false

Attribute	Description
pool-name prefix-delegation-pool-name	The pool-name attribute (optional) allows to assign of the IPv6 prefix from a locally managed pool. For more details, see Address Pool Configuration. This prefix is advertised by ICMPv6 router advertisements to the client where prefixes from optional prefix-delegation-pool-name are advertised by DHCPv6 as delegated prefix (IA_PD).
primary-dns secondary-dns	The primary DNS and secondary DNS servers configured are used by protocols such as ICMPv6 router advertisements or DHCPv6 and advertise to the client.
dad-enable	Enable (or disable) IPv6 duplicate address detection. **Default:** true

Enabling IPv6 Router Advertisement

IPv6 Router Advertisement (RA) is a component of the IPv6 Neighbor Discovery Protocol (NDP). It is used by routers to broadcast their availability and provide various network parameters to devices that support IPv6 on the network. Enabling Router Advertisement is crucial for the automatic configuration of IPv6 addresses and other settings on IPv6-enabled devices.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol ra
  <cr>
  enable                  Enable IPv6 router-advertisement
  interval                Interval
  lifetime                Lifetime
  preferred-lifetime      Preferred lifetime
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable IPv6 router advertisement. **Default:** false
interval	IPv6 router advertisements interval in seconds. Sets the interval at which the Router Advertisement messages are sent. **Default:** 0 (disabled)

Attribute	Description
lifetime	The valid lifetime for the prefix in seconds. **Default:** 14400
preferred-lifetime	The preferred lifetime for the prefix in seconds. **Default:** 1800

Enabling DHCPv4

To enable and configure DHCPv4 on the access profile, use the following command. This setup allows the access profile to handle DHCP requests and provide IP addresses to clients.

```
supervisor@switch: cfg> set access access-profile ipoe-dual protocol dhcp
  <cr>
  enable                      Enable DHCP
  lease-time                  DHCP lease time in seconds
  mode                        DHCP mode
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable DHCP. **Default:** false
dhcp-mode	This option specifies the DHCP mode for handling DHCP requests. **Default:** server **Values:** server and relay
lease-time	Define the duration of IP address leases, in seconds. **Default:** 300
dhcp-server	Configure global DHCP server.

Configuring DHCPv6

To configure DHCPv6 on an access profile, use the following command. This setup enables DHCPv6 functionality, prefix lifetimes, and the operating mode.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol dhcpv6
  <cr>
  enable                 Enable DHCPv6
  lifetime               Lifetime
  preferred-lifetime     Preferred lifetime
  mode                   DHCPv6 mode
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable DHCPv6. **Default:** false
mode	This option defines the DHCPv6 mode where the server handles DHCPv6 requests locally and relay/proxy forwards those to the configured servers. The difference between relay and proxy is that proxy can hide the actual DHCPv6 server. **Default:** server **Values:** server and relay
lifetime	The duration for which the advertised prefix is valid. This is the total time the prefix remains usable before it needs to be renewed. Set this to the desired lifetime in seconds. **Default:** 14400
preferred-lifetime	Set the lifetime for IPv6 prefixes, in seconds. It defines the duration during which the prefix is preferred for use. It should be less than or equal to the lifetime. If set higher than the lifetime, it will be adjusted to match the lifetime. The values for T1 and T2 are 0.5 and 0.8 times the shortest preferred lifetime. **Default:** 1800
dhcpv6-server	Configure DHCPv6 server.

Configuring PPPoE and PPP

The PPPoE protocol must be explicitly enabled in the access profile to allow PPPoE sessions.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol pppoe enable
true
```

PPPoE

The PPPoE configuration allows changing the default behavior of the PPPoE protocol.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol pppoe
  <cr>
  enable                 Enable PPPoE
  max-outstanding        Maximum outstanding PPPoE sessions
  session-protection     PPPoE session protection
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable PPPoE. **Default:** false
max-outstanding	Maximum outstanding PPPoE sessions. **Default:** 64 **Range:** 1 - 65535

When PPPoE session protection is enabled, any short-lived or failed sessions are logged. By default, a session that does not remain established for at least 60 seconds (min-uptime) is classified as a failed or short-lived session. Such failures trigger a block on new sessions for this IFP and VLAN for one second (min-lockout) by default. With each subsequent failed session, the lockout duration increases exponentially until it reaches a maximum of 300 seconds (max-lockout). If no failed sessions occur for 900 seconds, the lockout interval is reset (currently, not configurable).

PPPoE session protection also logs the last subscriber ID and terminates the session with a code that indicates the reason for the failure.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol pppoe
session-protection
  <cr>
  enable                 Enable PPPoE session protection
  max-lockout            Session protection maximum lockout time in seconds
  min-lockout            Session protection minimum lockout time in seconds
  min-uptime            Session protection minimum uptime in seconds
```

Attribute	Description
enable	Enables PPPoE session protection. **Default:** false
min-lockout	Session protection minimum lockout time (in seconds). **Default:** 1
max-lockout	Session protection maximum lockout time (in seconds). **Default:** 300
min-uptime	Session with an uptime less than this will trigger protection (in seconds). **Default:** 60

Configuring PPP LCP

The PPP Link Control Protocol (LCP) configuration allows changing the default behavior of the LCP protocol.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol ppp lcp
  <cr>
  authentication-protocol  Authentication protocol
  config-nak-max           Max configure-reject/nak [Range: <1-255>]
  echo-enable              Enable echo requests
  echo-interval            Echo interval in seconds [Range: <1-255>]
  echo-max-retransmit      Echo maximum retries [Range: <1-255>]
  lcp-loop-detection       Loop detection
  mru                      Maximum local MRU [Range: <1280-1500>]
  mru-negotiation          MRU negotiation
  mtu                      Maximum local MTU [Range: <1280-1500>]
  retransmit-interval      Retransmit interval in seconds [Range: <1-255>]
  retransmit-max           Maximum retries [Range: <1-255>]
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
authentication-protocol	This option allows you to specify the authentication protocol used during the LCP negotiation phase. By default, PPP authentication is set to PAP_CHAP. This can be changed by setting the authentication protocol to either PAP, CHAP, PAP_CHAP or CHAP_PAP. The Password Authentication Protocol (PAP) is (defined in RFC 1334) and receives the password as a plain text value from the client. The Challenge Handshake Authentication Protocol (CHAP) is (defined in RFC 1994) provides a more secure way to authenticate the client without exchanging plaintext secrets. The PAP_CHAP option first attempts to authenticate using PAP with a fallback to CHAP, if PAP is rejected by the client. Alternatively, CHAP_PAP starts with CHAP and falls back to PAP if CHAP is rejected by the client. **Default:** PAP_CHAP
echo-enable	By default, RBFS responds to LCP echo requests but does not initiate them unless echo-enable is set to true. **Default:** true
echo-interval	LCP echo request interval in seconds. **Default:** 30 **Range:** 1 - 255
echo-max-retransmit	LCP echo request retransmissions. **Default:** 3 **Range:** 1 - 255
mru-negotiation	Negotiate MRU **Default:** true
mru	Maximum local MRU (peer MTU) **Default:** 1492 **Range:** 1280 - 1500
mtu	Maximum local MTU (peer MRU) **Default:** 1492 **Range:** 1280 - 1500

Attribute	Description
lcp-loop-detection	This is used during the PPP connection process to detect and prevent loops in the network. It is achieved through the negotiation and validation of 'magic numbers'. Magic numbers are unique values used by both ends of a PPP connection to ensure that the data sent is not simply being looped back from the other end. By default, the negotiation and validation of magic numbers are enabled, providing protection against looping connections. You can disable loop detection by setting lcp-loop-detection to false. However, it is NOT recommended to disable LCP loop detection. **Default:** true
retransmit-interval	The LCP request retransmission interval. **Default:** 5 **Range:** 1 - 255
retransmit-max	The LCP requests retransmission before the session is terminated if no response is received. **Default:** 3 **Range:** 1 - 255
config-nak-max	The option config-nak-max defines the maximum PPP LCP configuration reject/nak messages that can be sent or received before the session is terminated. **Default:** 16 **Range:** 1 - 255

Configuring PPP IPCP

To enable IPv4 over PPPoE, both the address-family IPv4 and the protocol PPP IPCP must be explicitly enabled in the access profile. Additionally, the source-ifl option is a mandatory configuration that specifies the logical interface from which the local IPv4 address will be derived.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol ppp ipcp
  <cr>
  config-nak-max        Max configure-reject/nak [Range: <1-255>]
  enable                Enable PPP IPCP
  passive               Passive mode
  retransmit-interval   Retransmit interval in seconds [Range: <1-255>]
  retransmit-max        Maximum retries [Range: <1-255>]
  source-ifl            Source IFL
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable IPCP **Default:** false
passive	IPCP passive mode **Default:** false
source-ifl	This setting is required and should be set to obtain the local IPv4 address from a specific logical interface. It should be configured to use the loopback interface of the corresponding routing instance. If a source-ifl is chosen from a different routing instance, it will not change the routing instance assigned to the subscriber. Also, the routing instance can be changed through a RADIUS Access-Accept message without affecting the source-ifl. This means that the source-ifl can still be used to obtain the local IPv4 address, even if it belongs to a different routing instance. Although, it is not mandatory, it is recommended to configure the same loopback address across all routing instances where a subscriber might be located.
retransmit-interval	The IPCP request retransmission interval. **Default:** 5 **Range:** 1 - 255
retransmit-max	The IPCP requests retransmission before the session is terminated if no response is received. **Default:** 8 **Range:** 1 - 255
config-nak-max	The option config-nak-max defines the maximum PPP IPCP configuration reject/nak messages that can be sent or received before the session is terminated. **Default:** 8 **Range:** 1 - 255

Configuring PPP IP6CP

To use IPv6 over PPPoE, you must explicitly enable both the IPv6 address family and the PPP IP6CP protocol.

The following command demonstrates how to enable PPP IP6CP in the pppoe-dual access profile.

```
supervisor@switch: cfg> set access access-profile pppoe-dual protocol ppp ip6cp
  <cr>
  config-nak-max        Max configure-reject/nak [Range: <1-255>]
  enable                Enable PPP IP6CP
  passive               Passive mode
  retransmit-interval   Retransmit interval in seconds [Range: <1-255>]
  retransmit-max        Maximum retries [Range: <1-255>]
```

The following table provides the command options/attributes and descriptions.

Attribute	Description
enable	Enable IP6CP. **Default:** false
passive	IP6CP passive mode. **Default:** false
retransmit-interval	This option sets the interval, in seconds, between retransmissions, for IP6CP requests. **Default:** 5 **Range:** 1 - 255
retransmit-max	This option sets the maximum number of retransmission attempts. The IP6CP requests retransmission before the session is terminated if no response is received. **Default:** 8 **Range:** 1 - 255
config-nak-max	The option defines the maximum PPP IP6CP configuration reject/nak (Negative Acknowledgement) messages that can be sent or received before the session is terminated. **Default:** 6 **Range:** 1 - 255

DHCP Server Configuration

You can configure the DHCP server for IP address allocation using the following command and options.

Configuring DHCPv4 Server

The following commands and options are used to configure a DHCPv4 server.

```
supervisor@switch: cfg> set access dhcp-server
  dhcp-server           Global DHCP server configuration

supervisor@switch: cfg> set access dhcp-server server-example
  <cr>
  address               DHCP server address
  routing-instance      Instance name from which DHCP server is reachable
  source-address        Source address used for DHCP packets
```

The following example shows a DHCPv4 server configuration. It indicates the DHCP server name as 'dhcp-server1', which is the identifier for the DHCP server. The DHCP server’s IP address is specified as '172.16.0.10', and the source IP address is specified as '172.16.0.2', which is used by the DHCP server when sending out packets. The routing instance name is set as 'default', which indicates that the DHCP server is accessible through the default instance.

```
supervisor@switch: cfg> show config access dhcp-server
{
  "rtbrick-config:dhcp-server": [
    {
      "server-name": "dhcp-server1",
      "address": "172.16.0.10",
      "source-address": "172.16.0.2",
      "routing-instance": "default"
    }
  ]
}
```

Configuring DHCPv6 Server

The following commands and options are used to configure a DHCPv6 server.

```
supervisor@switch: cfg> set access dhcpv6-server dhcpv6-example
  <cr>
  address               DHCP server address
  routing-instance      Instance name from which DHCP server is reachable
  source-address        Source address used for DHCP packets
```

The following example shows a DHCPv6 configuration. It shows the DHCPv6 server name is specified as ‘dhcpv6-server1’ which is the identifier for the DHCPv6 server. The IP address is specified as '172:16::2'. The source IP address is specified as '172:16::10', which is used by the DHCPv6 server when sending out packets. The routing instance name is set as 'default'. It indicates the DHCPv6 server is accessible through the default instance.

```
supervisor@switch: cfg> show config access dhcpv6-server
{
  "rtbrick-config:dhcpv6-server": [
    {
      "server-name": "dhcpv6-server1",
      "address": "172:16::2",
      "source-address": "172:16::10",
      "routing-instance": "default"
    }
  ]
}
```

AAA Profile Configuration

Table: global.access.aaa.profile.config

The AAA profile for subscriber access is a mandatory configuration that serves as a cornerstone for managing and tracking subscriber activities. It ensures that all subscribers are authenticated before accessing services, authorized to use those services, and accounted for billing and monitoring purposes.

The following diagram illustrates how the AAA profile is associated with various subscriber management tasks.

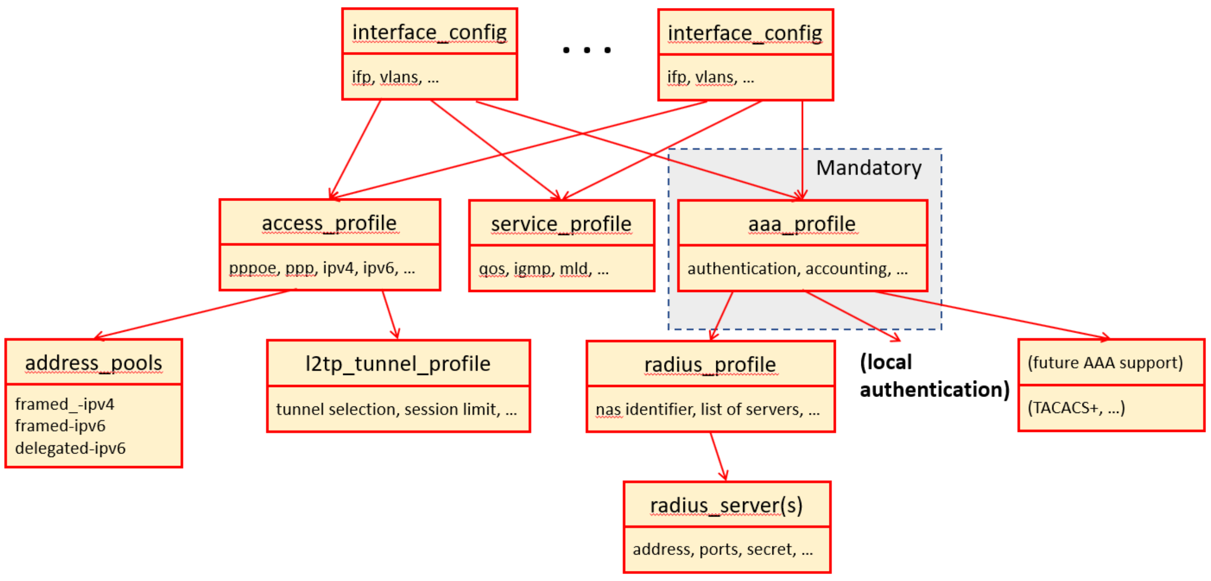


Figure 9. AAA Profile Configuration

Configuring the AAA Profile

The following command and options allow you to configure an AAA profile.

```
supervisor@switch: cfg> set access aaa-profile
  <profile-name>        Name of the AAA profile

supervisor@switch: cfg> set access aaa-profile aaa-example
  <cr>
  aaa-radius-profile    AAA RADIUS profile name
  accounting            Accounting options
  authentication        Authentication options
  idle-timeout          Idle timeout in seconds (0 == infinity) [Range: <0-
4294967295>]
  session-timeout       Session timeout in seconds (0 == infinity) [Range: <0-
4294967295>]
```

This following example demonstrates a typical AAA (Authentication, Authorization, and Accounting) profile configuration for RADIUS authentication and accounting. The profile name is 'aaa-radius.

The session-timeout value is set to '0', meaning there is no session timeout and the session can remain active indefinitely. The idle timeout value is also '0', indicating no idle timeout, so the session will not be terminated due to inactivity.

The parameter aaa-radius-profile is set to 'radius-default', which maps the AAA profile to the specific RADIUS profile named 'radius-default'. The RADIUS profile contains the necessary configuration details for communicating with the RADIUS server. In this profile, RADIUS is the selected method for handling all authentication requests. The order parameter indicates that RADIUS is the primary and only method of authentication configured.

For accounting, the order is specified as RADIUS, indicating that all accounting data is sent to the RADIUS server. The parameter session-id-format is set to "DEFAULT".

The ingress accounting-source is "POLICER", and the egress accounting source is "CLASS". "POLICER" indicates that the accounting data is based on a traffic policer, while "CLASS" indicates that the accounting data is based on traffic classes.

The class-byte-adjustment-value parameter is set to 16, adjusting the byte count for outgoing traffic by a specific value (16).

```
supervisor@switch: cfg> show config access aaa-profile aaa-radius
{
  "rtbrick-config:aaa-profile": {
    "profile-name": "aaa-radius",
    "session-timeout": 0,
    "idle-timeout": 0,
    "aaa-radius-profile": "radius-default",
    "authentication": {
      "order": "RADIUS"
    },
    "accounting": {
      "order": "RADIUS",
      "session-id-format": "DEFAULT",
      "ingress": {
        "accounting-source": "POLICER"
      },
      "egress": {
        "accounting-source": "CLASS",
        "class-byte-adjustment-value": 16
      }
    }
  }
}
```

Attribute	Description
session-timeout	The session timeout defines the maximum (uptime) duration, in seconds, that a subscriber can remain connected before the session is automatically terminated. The value '0' indicates that the session has no time limit and can continue indefinitely. **Default:** 0 **Range:** 0 - 4294967295
idle-timeout	The idle timeout is the time, in seconds, after which a subscriber will be disconnected if no data is being transmitted through their outgoing logical interface (IFL). Control traffic is not included in these statistics. The subscriber is not considered idle as long as outgoing traffic is detected. The idle timeout is not limited but should be set to at least double the time of the logical interface statistics counter update interval (between 5 to 45 seconds). The idle timeout does not have a specific limit, but it is recommended to set it to at least double the time of the logical interface statistics counter update interval (which is between 5 to 45 seconds). Therefore, it is advisable to set a minimum idle timeout of 90 seconds. The value '0' means that the timeout is infinite. **Default:** 0 **Range:** 0 - 4294967295

Attribute	Description
aaa-radius-profile	The RADIUS profile, (for more details, see RADIUS Profile Configuration), which is used if RADIUS authentication or accounting is enabled.

Configuring Authentication

RBFS supports the authentication methods: NONE, LOCAL, and RADIUS.

- NONE: Disables authentication, accepting all credentials without verification.
- LOCAL: Authenticates the subscriber based on locally defined user profiles. For more details, refer to User Profile Configuration.
- RADIUS: Authenticates the subscriber remotely by sending an authentication request to the configured RADIUS servers.

The authentication method DOMAIN is currently not supported.

Certain authentication methods can be combined together such as the following:

LOCAL_RADIUS: In this method, the system attempts to authenticate the subscriber at first locally. If no matching local user is found, the system attempts authentication via RADIUS. If a local user is found but the password does not match, the subscriber is immediately rejected, and no RADIUS request is made.

RADIUS_LOCAL: In this method, the system attempts to authenticate the subscriber at first through RADIUS. If the RADIUS server rejects the request, the subscriber is immediately disconnected. However, if no response (timeout) is received from any configured RADIUS server, the system falls back to local authentication.

The following command and options allow you to define the order of authentication methods.

```
supervisor@switch: cfg> set access aaa-profile aaa-default authentication
  <cr>
  delimiter                 Delimiter string
  order                     Authentication order
```

Attribute	Description
order	This option defines the order of authentication methods. **Default:** NONE **Values:** LOCAL, LOCAL_RADIUS, RADIUS, RADIUS_LOCAL
delimiter	This option defines the delimiter for domain authentication. **Default:** @ Currently, the delimiter option is not supported.

Configuring Accounting

Subscriber accounting involves tracking and recording a subscriber’s session duration and data usage. This process includes two main components:

- Time Accounting: Measures the duration of a subscriber’s session.
- Volume Accounting: Tracks the number of packets and bytes transmitted or received.

Volume accounting operates in both directions (ingress and egress), and these can be configured independently.

Currently, RBFS supports RADIUS as the only accounting method.

The following command and options are used to configure accounting for the default AAA profile named aaa-default. The 'egress' option allows you configure accounting for outgoing traffic, and the 'ingress' option allows to set accounting for incoming traffic. The 'interim-interval' option sets the interval, in seconds, for sending interim accounting updates. The 'order' option determines the sequence in which accounting records are processed or sent. This is helpful when multiple accounting methods are used. The 'session-id-format' option allows you to define the format of the accounting session ID attribute, which uniquely identifies each accounting session.

```
supervisor@switch: cfg> set access aaa-profile aaa-default accounting
  <cr>
  egress                 Egress volume accounting options
  ingress                Ingress volume accounting options
  interim-interval       Accounting interim interval in seconds (0 == disabled)
[Range: <0-4294967295>]
  order                  Accounting order
  session-id-format      Accounting-Session-Id format
```

Attribute	Description
order	This option defines the order of accounting methods. **Default:** NONE
interim-interval	The interim interval specifies the time between interim accounting requests in seconds where 0 means disabled. **Default:** 0 **Range:** 0 - 4294967295
session-id-format	The format of the Accounting-Session-Id (RADIUS attribute 44). | Name | Format | Example | |---|---|---| | DEFAULT | <subscriber-id>.<timestamp> | 72339069014639577.1551943760 | | BRIEF | <subscriber-id>> | 72339069014639577 | | EXTENSIVE | <subscriber-id>.<ifp>.<outer-vlan>.<inner-vlan>.<client-mac>.<session-id>.<timestamp> | 72339069014639577.ifp-0/0/0.128.7.01:02:03:04:05:05.1.1551943760 | **Default:** DEFAULT **Values:** BRIEF, EXTENSIVE Currently, only DEFAULT is supported.

Configuring Accounting Adjustments

The accounting adjustment feature enables basic counter modifications for the configured accounting method, such as RADIUS accounting. This configuration is necessary to normalize counters across different platforms in each direction. On Broadcom Q2C and Q2A based platforms, packets are counted in the size they enter the switch. Without adjustment, egress accounting would count downstream traffic as received from the core, complete with MPLS labels, while ingress accounting typically includes VLAN headers and/or PPPoE headers.

This counter adjustment aims to normalize counters with diverse encapsulations (double-tagged, untagged, and so on), potentially aligning to L3 counters (IP header and payload) as an example, or exclusively adapting egress traffic to match the outgoing packet encapsulation. The possibility for separate adjustment configurations per direction allows parity in the counters for both ingress and egress.

Example Scenario:

In a scenario where packets enter a device with various encapsulations like PPPoE headers, MPLS labels, or VLAN tags, the system may count these additional headers in the packet size if the adjustment feature is not enabled. This can cause discrepancies in accounting reports.

For instance, an egress interface where traffic exits the device and goes to subscribers. If this traffic originally came from the core network with MPLS labels attached, the system counts the bytes of those labels in its accounting, inflating the byte count. Similarly, on the ingress side, the device counts VLAN tags or PPPoE headers, further impacting accurate accounting reports. To address this issue, you can configure a byte adjustment on both the ingress and egress sides to normalize these counters.

In RBFS, two configuration options are available for to achieve this: the byte adjustment value and the byte adjustment factor. The option byte adjustment value, which can be positive or negative (for example, -20 or 20), adjusts the byte count by a specific amount. Any decimal digits in the adjustment value are rounded to the nearest whole number (for example, 20.2 becomes 20). The option byte adjustment factor, which is less commonly used, adjusts the byte count by a percentage. It accepts positive values and considers up to two decimal places, such as 0.98 for a 2% reduction or 1.02 for a 2% increase.

Ingress Accounting

Subscriber ingress accounting refers to the process of measuring and recording the data usage or traffic that enters a subscriber interface (upstream).

The following command and options allow you to set data usage accounting for incoming traffic.

```
supervisor@switch: cfg> set access aaa-profile aaa-default accounting ingress
  <cr>
  accounting-source                Source of session ingress counter
  byte-adjustment-factor           Adjust ingress IFL counters by factor (executed
after adjustment value) [Range: <0.00-2.00>]
  byte-adjustment-value            Adjust ingress IFL counters by N bytes per
packet [Range: <-32-32>]
  policer-byte-adjustment-factor  Adjust ingress policer counters by factor
(executed after adjustment value) [Range: <0.00-2.00>]
  policer-byte-adjustment-value   Adjust ingress policer counters by N bytes per
packet [Range: <-32-32>]
```

Attribute	Description
accounting-source	This option provides control over the counters used for subscriber ingress accounting when RADIUS accounting is enabled. The counters in question are the RADIUS attributes Acct-Input-Packets (47), Acct-Input-Octets (42), and Acct-Input-Gigawords (52). By default, the policer statistics (POLICER) are utilized, which represent the total traffic accepted across all policer levels (1-4). However, ingress control traffic is subject to a separate control plane policer and is therefore not included in the session policer statistics. Consequently, policers are necessary if session accounting is required. Alternatively, the logical interface (IFL) statistics can be used, which capture all received traffic, including control traffic and the traffic that is dropped by the ingress policer. It is important to note that this option may NOT be available on all platforms. **Default:** POLICER **Values:** POLICER, IFL
byte-adjustment-value	Adjusts ingress IFL counters by +/- N bytes per packet. It allows you to adjust the ingress IFL counters by adding or subtracting a specified number of bytes per packet. **Default:** 0.00 **Range:** -32 to 32
byte-adjustment-factor	Adjust the ingress IFL counters by applying a specified factor. (This adjustment is made after the byte-adjustment-value is applied). **Default:** 1.00 **Range:** 0.00 - 2.00

Attribute	Description
policer-byte-adjustment-value	Adjusts ingress POLICER counters by +/- N bytes per packet. It adjusts the ingress POLICER counters by adding or subtracting a specified number of bytes per packet. **Default:** 0.00 **Range:** -32 to 32
policer-byte-adjustment-factor	Adjusts the ingress POLICER counters by applying a specified factor. This adjustment is also executed after the policer-byte-adjustment-value is applied. **Default:** 1.00 **Range:** 0.00 - 2.00

Egress Accounting

Subscriber egress accounting refers to the process of measuring and recording the data usage or traffic that is sent from a subscriber interface (downstream).

The following command and options allow you to set data usage accounting for outgoing traffic.

```
supervisor@switch: cfg> set access aaa-profile aaa-default accounting egress
  <cr>
  accounting-source              Source of session egress counter
  byte-adjustment-factor         Adjust egress IFL counters by factor (executed
after adjustment value) [Range: <0.00-2.00>]
  byte-adjustment-value          Adjust egress IFL counters by N bytes per packet
[Range: <-32-32>]
  class-byte-adjustment-factor   Adjust egress class counters by factor (executed
after adjustment value) [Range: <0.00-2.00>]
  class-byte-adjustment-value    Adjust egress class counters by N bytes per packet
[Range: <-32-32>]
```

Attribute	Description
accounting-source	This option provides control over the counters used for egress session accounting when RADIUS accounting is enabled. The counters are the RADIUS attributes: Acct-Output-Packets (48), Acct-Output-Octets (43), and Acct-Output-Gigawords (53). By default, the class statistics (CLASS) are used, which represent the total traffic accepted across all queues. However, the egress control traffic is sent directly to the IFP and is, therefore, not included in the session class statistics. As a result, QoS is necessary for enabling the egress session accounting. As an alternative, the logical interface (IFL) statistics can be used to account for all transmitted traffic, excluding control traffic. However, this option may not be supported on all platforms. **Default:** CLASS **Values:** CLASS, IFL
byte-adjustment-value	Adjust egress IFL counters by +/- N bytes per packet. That is, adjust egress IFL counters by adding or subtracting a specified number of bytes per packet. **Default:** 0.00 **Range:** -32 to 32
byte-adjustment-factor	Adjust egress IFL counters by applying a specified factor. This adjustment is made after the byte-adjustment-value is applied. **Default:** 1.00 **Range:** 0.00 to 2.00
class-byte-adjustment-value	Adjusts egress CLASS (queue) counters by adding or subtracting a specified number of bytes per packet (+/- N bytes per packet). **Default:** 0.00 **Range:** -32 to 32

Attribute	Description
class-byte-adjustment-factor	Adjust egress CLASS (queue) counters by applying a specified factor. This adjustment is also made after the class-byte-adjustment-value is applied. **Default:** 1.00 **Range:** 0.00 to 2.00

RADIUS Profile Configuration

Subscriber management allows the configuration of a RADIUS profile, which is mandatory if RADIUS is used for authentication or accounting.

The diagram below illustrates how the RADIUS profile is associated with the subscriber management tasks at a broader level.

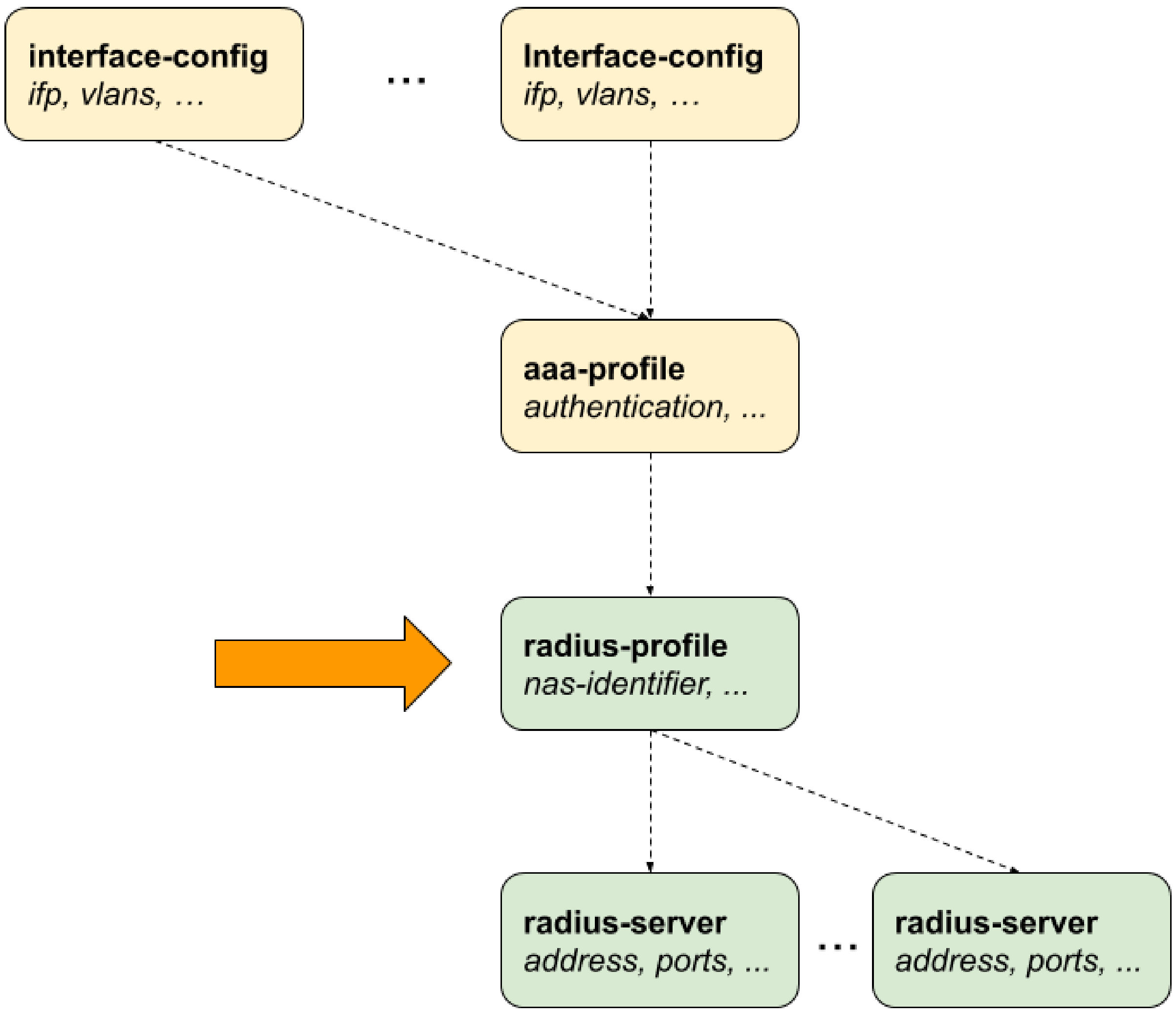


Figure 10. RADIUS Profile Configuration

Configuring the RADIUS Profile

The RADIUS profile configuration involves setting up various parameters that define how the router interacts with the RADIUS server for authentication, accounting, and so on.

```
supervisor@switch: cfg> set access radius-profile
  <profile-name>        Name of the RADIUS profile

supervisor@switch: cfg> set access radius-profile radius-default
  <cr>
  accounting            RADIUS accounting options
  authentication        RADIUS authentication options
  nas-identifier        NAS identifier
  nas-ip-address        NAS IP address (IPv4 Address)
  nas-port-format       NAS-Port format
  nas-port-type         NAS-Port type
```

The following example shows a typical RADIUS profile for authentication and accounting. The RADIUS profile name is specified as 'radius-default'. NAS Identifier is set as BNG and NAS Port Type is specified as 'Ethernet'. RADIUS servers used for authentication is set 'radius-server-1' and 'radius-server-2'. RADIUS servers used for accounting is also set as 'radius-server-1' and 'radius-server-2'.

The Accounting RADIUS servers are specified as 'radius-server-1' 'radius-server-2'. For information about other options, see the table descriptions below.

```
supervisor@switch: cfg> show config access radius-profile radius-default
{
  "rtbrick-config:radius-profile": {
    "profile-name": "radius-default",
    "nas-identifier": "BNG",
    "nas-port-type": "Ethernet",
    "authentication": {
      "radius-server-profile-name": [
        "radius-server-1",
        "radius-server-2"
        ]
    },
    "accounting": {
      "radius-server-profile-name": [
        "radius-server-1",
        "radius-server-2"
        ],
      "stop-on-reject": "true",
      "stop-on-failure": "true",
      "accounting-on-off": "true",
      "accounting-on-wait": "true",
      "accounting-backup": "true",
      "accounting-backup-max": 86400
    }
  }
}
```

Attribute	Description
nas-identifier	Set the value for the RADIUS attribute NAS-Identifier (32). **Default:** system hostname
nas-ip-address	Set the value for RADIUS attribute NAS-IP-Address (4). **Default**: source IPv4 address
nas-port-type	Set the value for RADIUS attribute NAS-Port-Type (61). **Default:** Ethernet
nas-port-format	Set the format of the 32-bit RADIUS attribute NAS-Port (5). **Name** / **Bits** / **Values** DEFAULT / 1:1:6:12:12 / slot:subslot:port:vlan:vlan SLOTS / 6:2:6:12:6 / slot:subslot:port:vlan:vlan PORTS / 0:1:7:12:12 / slot:subslot:port:vlan:vlan

Configuring Authentication

The following configuration command and options set the 'radius-profile' as authentication method. It also defines the list of RADIUS servers and the algorithm for RADIUS high-availability.

```
supervisor@switch: cfg> set access radius-profile radius-default authentication
  <cr>
  algorithm-type               Authentication redundancy algorithm
  radius-server-profile-name   RADIUS server profile name
```

Attribute	Description
radius-server-profile-name	List of multiple RADIUS servers used for authentication.
algorithm-type	Specifies the authentication server selection algorithm. For more information, see RADIUS Redundancy. **Default:** DIRECT **Values:** DIRECT, ROUND-ROBIN

Configuring Accounting

Configuring accounting involves specifying its name and defining various parameters.

The following command and options allow you to configure RADIUS accounting.

```
supervisor@switch: cfg> set access radius-profile radius-default accounting
  <cr>
  accounting-backup            Enables backup accounting
  accounting-backup-max        Maximum backup accounting hold time, in seconds
  accounting-on-off            Enables accounting on/off
  accounting-on-wait           Wait for an accounting-on response before sending
authentication requests
  algorithm-type               Accounting redundancy algorithm
  radius-server-profile-name   RADIUS server profile name
  stop-on-failure              Send accounting-stop on failure
  stop-on-reject               Send accounting-stop on authentication reject
```

Attribute	Description
radius-server-profile-name	List of RADIUS servers used for accounting.
algorithm-type	Specifies the accounting server selection algorithm. For more information, see RADIUS Redundancy. **Default:** DIRECT **Values:** DIRECT, ROUND-ROBIN
stop-on-failure	If set to true, the accounting will stop, if there is a failure in the process after authentication was accepted. **Default:** false
stop-on-reject	If set to true, when the authentication is rejected, the accounting process will stop. **Default:** false
accounting-on-off	Enables RADIUS Accounting-On/Off messages. For more information, see RADIUS Accounting. **Default:** false

Attribute	Description
accounting-on-wait	Waits for an Accounting-On response ensuring that no new subscriber is allowed until the accounting process has been initiated. **Default:** false
accounting-backup	Enables backup for accounting (optional). RADIUS accounting requests are often used for billing and, therefore should be able to store and retry over a longer period (commonly, up to 24 hours or more). **Default:** false
accounting-backup-max	If enabled, this option defines maximum backup accounting hold time, in seconds. **Default:** 3600 **Range:** 1 - 4294967295

RADIUS Server Configuration

Subscriber Management often relies on the RADIUS server for authentication, authorization, and accounting. There are alternative AAA solutions, including local methods that operate independently of network availability.

The RADIUS server configuration is not standalone; it is inherently dependent on the configuration of a RADIUS profile. You must first create a RADIUS profile before configuring the associated RADIUS server.

The following diagram illustrates how RADIUS server configuration fits into the broader context of subscriber management tasks.

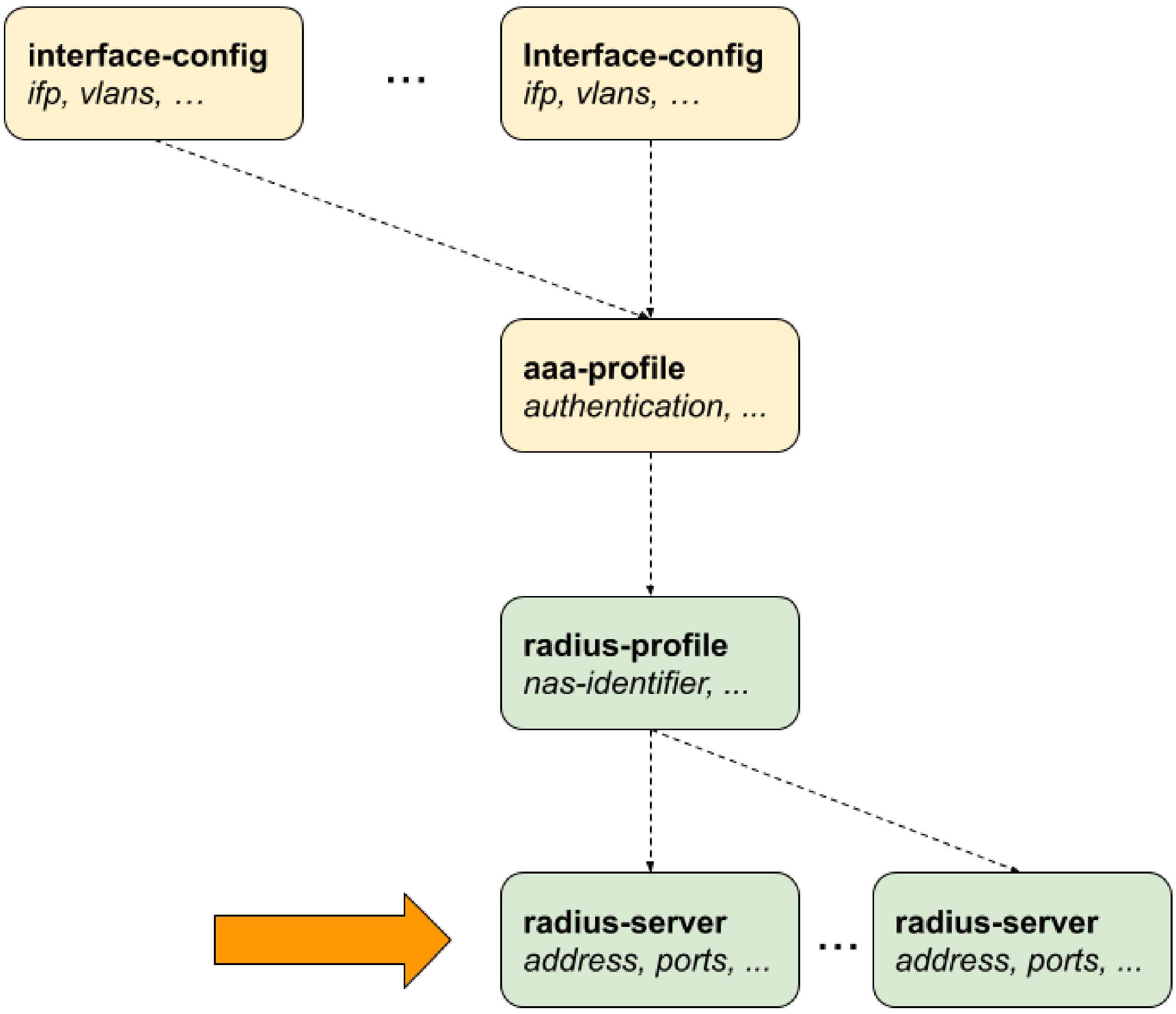


Figure 11. RADIUS Server Configuration

Configuring the RADIUS Server

The following configuration command and options are used to configure a RADIUS server.

```
supervisor@switch: cfg> set access radius-server
  <server-name>          Name of the RADIUS server

supervisor@switch: cfg> set access radius-server radius-server-1
  <cr>
  accounting              RADIUS accounting mode
  address                 RADIUS server address
  authentication          RADIUS authentication mode
  coa                     RADIUS Change-of-Authorization (CoA) mode
  rate                    Maximum RADIUS requests per/second
  routing-instance        Instance name
  secret-encrypted-text   RADIUS secret in encrypted text
  secret-plain-text       RADIUS secret in plain text
  source-address          Source address used for RADIUS packets
```

The following example shows the RADIUS server configuration for authentication and accounting. Each of these options allows you to define how the RADIUS server will operate. The command sets RADIUS server name as ‘radius-server-1’. The configuration specifies the IP address '198.51.100.101' of the RADIUS server, which the router will use to send RADIUS requests. The source IP address is specified as '198.51.100.200'. It sets 'CoA' (Change-of-Authorization), a feature that allows the RADIUS server to change the authorization parameters of a session after it has been established.

The secret-encrypted-text parameter is specified that configures the RADIUS secret in an encrypted format. The key is used for encrypting RADIUS messages between the client and server to ensure security.

```
supervisor@switch: cfg> show config access radius-server radius-server-1
{
  "rtbrick-config:radius-server": {
    "server-name": "radius-server-1",
    "address": "198.51.100.101",
    "source-address": "198.51.100.200",
    "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
    "authentication": {
      "enable": "true"
    },
    "accounting": {
      "enable": "true"
    },
    "coa": {
      "enable": "true"
    }
  }
}
```

Attribute	Description
address	RADIUS server IPv4 address. *Multiple RADIUS servers with the same IPv4 address are currently not supported, even if the instance or port is different.!*
source-address	Local source IPv4 address.
routing-instance	The routing instance in which the RADIUS server is reachable.
secret-encrypted-text secret-plain-text	RADIUS secret, which can be provided as plaintext or already encrypted text.

Attribute	Description
rate	Maximum RADIUS requests per second. **Default:** 600 **Range:** 1 - 65535

Configuring Authentication

The following command and options allow to configure the authentication settings on a RADIUS server. The enable option is used to activate RADIUS authentication for the specified RADIUS server 'radius-server-1'. The outstanding option allows you to specify the maximum number of pending authentication requests that the device can have at any given time. The port option enables you to define the port number used for RADIUS authentication requests. The retry option defines the maximum number of retries the device makes if the RADIUS server does not acknowledge an authentication request. This ensures that transient network issues do not cause authentication failures. The timeout setting determines the time, in seconds, that the device will wait for a response from the RADIUS server after sending authentication requests. If the server does not respond within this time frame, the request will be considered timed out.

```
supervisor@switch: cfg> set access radius-server radius-server-1 authentication
  <cr>
  enable                      Enable RADIUS authentication
  outstanding                 Maximum number of outstanding authentication requests
  port                        RADIUS server authentication port
  retry                       Maximum retries for authentication request packets
  timeout                     Authentication request timeout in seconds
```

Attribute	Description
enable	Enables RADIUS authentication. **Default:** false
port	RADIUS authentication port. **Default:** 1812 **Range:** 1 - 65535

Attribute	Description
retry	This option specifies the number of authentication retries before declaring this server as unreachable for authentication. After reaching the limit, the client begins to send requests to other RADIUS servers and rejects the request after receiving the end of the list. **Default:** 3 **Range:** 1 - 255
timeout	Authentication request timeout, in seconds. **Default:** 5 **Range:** 1 - 65535
outstanding	This option specifies the maximum number of outstanding authentication requests for this RADIUS server. A request is counted as outstanding if sent out but the response is not received. **Default:** 100 **Range:** 1 - 65535

Configuring Accounting

RADIUS accounting allows the tracking of user sessions. This configuration sets up RADIUS accounting on a specified RADIUS server.

When enabled, the device will send accounting data to the RADIUS server.

```
supervisor@switch: cfg> set access radius-server radius-server-1 accounting
  <cr>
  enable                      Enable RADIUS accounting
  outstanding                 Maximum number of outstanding accounting requests
  port                        RADIUS server accounting port
  retry                       Maximum retries for accounting request packets
  timeout                     Accounting request timeout in seconds
```

Attribute	Description
enable	Enable RADIUS accounting. **Default:** false

Attribute	Description
port	This specifies the port number on the RADIUS server that is used for accounting purpose. **Default:** 1813 **Range:** 1 - 65535
retry	This option determines the maximum number of attempts the device will make to send an accounting request to the RADIUS server before marking the server as unreachable for accounting. Once the retry limit is reached, the device will attempt to send accounting requests to other configured RADIUS servers. **Default:** 10 **Range:** 1 - 255
timeout	Accounting request timeout, in seconds. **Default:** 30 **Range:** 1 - 65535
outstanding	This option specifies the maximum number of outstanding accounting requests for this RADIUS server. A request is counted as outstanding if sent out, but the response is not received. **Default:** 100 **Range:** 1 - 65535

Configuring Change-of-Authorization (CoA)

Change-of-Authorization (CoA) allows a RADIUS server to send requests to the network device to change the authorization parameters of an active session. The following configuration involves setting up CoA for a specified RADIUS server called 'radius-server-1'.

```
supervisor@switch: cfg> set access radius-server radius-server-1 coa
  <cr>
  enable                Enable Change-of-Authorization (CoA)
  port                  Local RADIUS CoA port
```

Attribute	Description
enable	This option enables the device to receive CoA requests from the specified RADIUS server. **Default:** false
port	Specifies the device port that listens/receives for CoA requests from the RADIUS server. **Default:** 3799 **Range:** 1 - 65535

Service Profile Configuration

You can set up Service Profile to assign specific Quality of Service (QoS) and IGMP settings to individual subscribers. This ensures that each subscriber gets the right level of service based on their requirements. The Service Profile configuration is optional.

The following diagram illustrates how the service profile is associated with other subscriber management tasks.

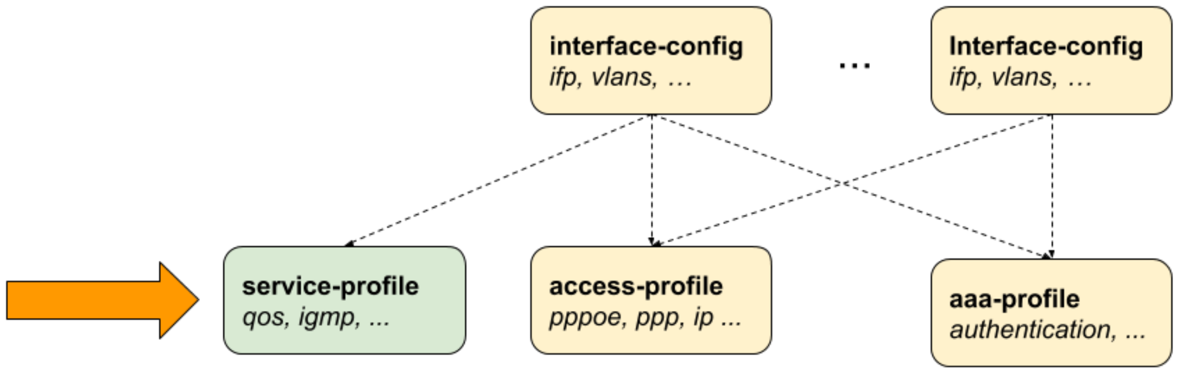


Figure 12. Service Profile Configuration

Configuring the Service Profile

The following command and options are used to set up a service profile named 'iptv'.

```
supervisor@switch: cfg> set access service-profile
  <profile-name>        Name of the service profile

supervisor@switch: cfg> set access service-profile iptv
  <cr>
  igmp                      IGMP related attributes
  qos                       QoS related attributes
```

The following example shows a typical service profile configuration for subscribers with IPTV (multicast) services. The configuration defines specific QoS and IGMP settings that will be applied to subscribers using this service profile. In this configuration, the service profile named ‘iptv’ is linked to a QoS profile named ‘iptv-qos-xl’. This QoS profile defines the prioritization and bandwidth allocation for IPTV traffic for the subscriber profile that it is associated with.

Additionally, the configuration specifies the IGMP settings. Enabling IGMP is achieved by setting it to 'true' for the service profile that enables subscribers to join multicast groups. The configuration specifies the IGMP profile named 'iptv-basic', with IGMP version 3 in use. The parameter max-members is set to 10, which limits the number of simultaneous multicast groups (maximum IGMP memberships) that a subscriber can join.

```
supervisor@switch: cfg> show config access service-profile iptv
{
  "rtbrick-config:service-profile": {
    "profile-name": "iptv",
    "qos": {
      "profile": "iptv-qos-xl"
    },
    "igmp": {
      "enable": "true",
      "profile": "iptv-basic",
      "version": "IGMPv3",
      "max-members": 10
    }
  }
}
```

Configuring QoS

The following QoS configuration defines the QoS settings for the service profile 'iptv'.

```
supervisor@switch: cfg> set access service-profile iptv qos
  <cr>
  parent-scheduler          QoS parent scheduler
  profile                   QoS profile
```

Attribute	Description
parent-scheduler	This option defines the parent scheduler element within the scheduler-map that is assigned to the subscriber. If the parent scheduler is not specified, the scheduler-map will be directly bound to the local IFP where the session is established. This attribute can only be set once and cannot be changed without disconnecting the session. The parent scheduler can also be configured through RADIUS, which gets priority over the local configuration. ⚠ Assigning a QoS parent scheduler that does not exist on the corresponding IFP will result in blackholing all egress data traffic. Control traffic will remain unaffected, allowing the session to persist despite the data loss.
profile	This option assigns a specific QoS configuration profile to the subscriber. The QoS profile can be also set through RADIUS. If configured both locally and through RADIUS, the RADIUS setting gets priority.

Configuring IGMP

The following IGMP configuration defines the settings for the service profile 'iptv'.

```
supervisor@switch: cfg> set access service-profile iptv igmp
  <cr>
  enable                      Enable IGMP service
  max-members                 Maximum IGMP membership per subscriber
  profile                     IGMP profile
  version                     IGMP version
```

Attribute	Description
enable	This option dynamically enables or disables IGMP for a subscriber. **Default:** false

Attribute	Description
max-members	This option limits the number of simultaneous multicast channels (maximum IGMP memberships) a subscriber can join. **Default:** 1 **Range:** 1 - 4294967295
profile	This option specifies the IGMP profile to be associated with the subscriber.
version	This defines the version of IGMP for a subscriber. **Default:** V3 **Values:** V1, V2, V3

L2TP Profile Configuration

The configuration of the Layer 2 Tunnel Protocol (L2TPv2) profile is optional for subscriber management. It is necessary only if you want to enable L2TP tunneling.

The following diagram illustrates how the L2TP profile configuration is are associated with the other Subscriber Management tasks.

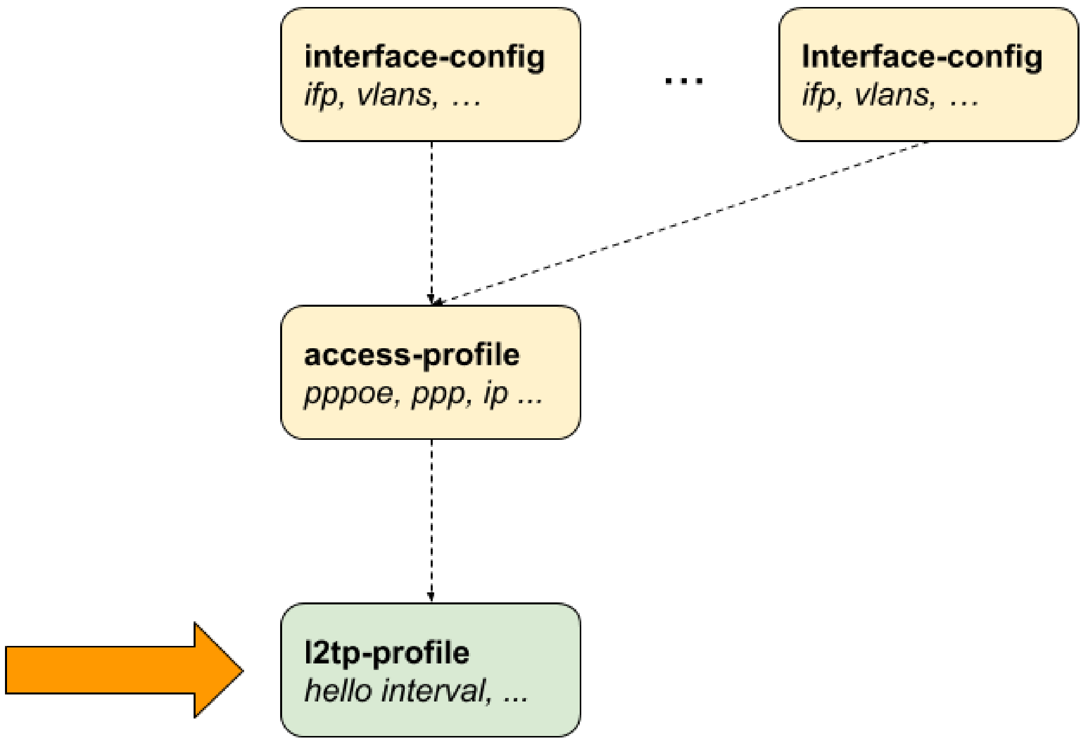


Figure 13. L2TPv2 Profile Configuration

Configuring the L2TP Profile

The following command and options allow you to configure an L2TP profile.

```
supervisor@switch: cfg> set access l2tp-profile
  <profile-name>        Name of the L2TP profile

supervisor@switch: cfg> set access l2tp-profile l2tp-default
  <cr>
  client-ipv4                Default value for L2TP tunnel client IPv4 address
  client-name                Default value for L2TP tunnel client name
  connect-speed-update       Enable L2TP Connect-Speed-Update-Notification (CSUN)
  dead-timeout-interval      L2TP tunnel dead timeout interval in seconds [Range:
<1-4294966>]
  hello-interval             L2TP tunnel hello interval in seconds [Range: <0-
86400>]
  hide-authentication        Hide L2TP tunnel authentication
  idle-timeout-interval      L2TP tunnel idle timeout interval in seconds [Range:
<0-4294966>]
  inactive-timeout-interval  L2TP tunnel inactive timeout interval in seconds
[Range: <1-4294966>]
  instance                   Instance name
  pon-access-line-version    PON Access Line Information Version
  pool-name                  L2TP tunnel pool name
  receive-window             L2TP tunnel receive window [Range: <1-256>]
  request-retries            L2TP session request retries [Range: <1-600>]
  request-timeout-interval   L2TP session request timeout interval in seconds
[Range: <1-30>]
  retransmit-interval        L2TP tunnel retransmission interval in seconds
[Range: <1-30>]
  selection-algorithm        L2TP tunnel selection algorithm
  service-label              MPLS service label
  session-limit              L2TP tunnel session limit [Range: <1-65535>]
```

The following example shows a typical L2TPv2 LAC configuration profile. It outlines a typical L2TPv2 LAC profile, defining essential parameters such as session limits, heartbeat intervals, client identification, and security settings.

This configuration example defines an L2TPv2 LAC (L2TP Access Concentrator) profile named 'l2tp-default'. The profile-name is set 'l2tp-default'. This session-limit parameter sets the maximum number of sessions that can be established within this L2TP tunnel. The value '4000' indicates that up to 4000 sessions are allowed. The hello-interval is set to '60' which indicates the interval (in seconds) for sending 'hello' messages to check the status of the L2TP tunnel.

The client-name is specified as 'BNG', which is the name of the client of the L2TP tunnel. The client-ipv4 is set to '198.51.100.200' which is the 'BNG' IP address. The hide-authentication parameter determines whether the authentication credentials for the L2TP tunnel should be hidden. Setting it to true indicates the credentials are hidden.

The parameter service-label is set to '1234'. It assigns an MPLS service label to the L2TP tunnel. The value '1234' is used as the service label for this tunnel.

```
supervisor@switch: cfg> show config access l2tp-profile l2tp-default
{
  "rtbrick-config:l2tp-profile": {
    "profile-name": "l2tp-default",
    "session-limit": 4000,
    "hello-interval": 60,
    "client-name": "BNG",
    "client-ipv4": "198.51.100.200",
    "hide-authentication": true
    "service-label": 1234
  }
}
```

Attribute	Description
client-ipv4	This is the default IPv4 address for the local L2TP tunnel client (LAC), if no specific address is provided through the L2TP pool or RADIUS configuration.
client-name	This is the default hostname for the local L2TP tunnel client (LAC) if no specific hostname is provided through the L2TP pool or RADIUS configuration. **Default:** system hostname
instance	The routing instance in which the L2TP endpoint (LNS) is reachable. **Default:** default
service-label	Define the service label to support L2TP over MPLS. For more details, see Configuring L2TP over MPLS. Supported MPLS label values range from 0 to 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. To avoid conflicts, it is recommended to assign label values outside of these reserved ranges.

Attribute	Description
selection-algorithm	It defines how to select a tunnel from a pool of available LNS servers. For more information, see L2TP Tunnel Selection. The RANDOM algorithm selects a tunnel randomly, while the BALANCED algorithm chooses the tunnel with the fewest sessions. **Default:**: BALANCED **Values:** BALANCED, RANDOM
session-limit	This is the default session limit for a tunnel unless otherwise specified. Tunnels that have reached their session limit are not considered for additional sessions. **Default:** 64000 **Range:** 1 - 65535
pool-name	This allows to assign a default L2TP tunnel pool. For more information, see L2TP Tunnel Pool Configuration. This default can be overwritten by user-defined pool names from the local user profiles (For more information, see User Profile Configuration) or received through RADIUS attribute RtBrick-L2TP-Pool (VSA 26-50058-40).
hello-interval	Specifies the L2TP tunnel hello interval time, in seconds. '0' indicates disabled. The HELLO keep-alive messages are part of the L2TP control channel. For more information, see L2TP Control Channel. These messages are sent only when the message queue is empty, and no other messages have been transmitted during the hello interval. **Default:** 30 **Range:** 0 - 86400
idle-timeout-interval	This interval defines the maximum time, in seconds, to keep a tunnel without sessions established. The session will remain forever if this value is set to 0. **Default:** 600 **Range:** 0 - 4294966

Attribute	Description
dead-timeout-interval	This interval defines the time, in seconds, This interval defines the time, in seconds, that a tunnel remains in the DEAD state if it becomes unreachable. Once the interval expires, the tunnel transitions back to the DOWN state, making it available for new sessions. **Default:** 300 **Range:** 1 - 4294966
inactive-timeout-interval	This interval defines the time, in seconds, to keep an inactive tunnel before removal. This interval is reset with every new session request which considers this tunnel as a potential candidate. **Default:** 900 **Range:** 1 - 4294966
receive-window	Specifies the receive window size offered to the remote peer trough the Receive Window Size AVP (10) in SCCRQ or SCCRP message. For example, if a receive window size of '8' is advertised in the SCCRQ or SCCRP message, the remote peer can send up to 8 unacknowledged control messages. After reaching this limit, it must wait for an acknowledgment to advance the window before sending additional messages. **Default:** 8 **Range**: 1 - 256
request-retries	Defines the number of times the system attempts to establish a tunnel before considering it unsuccessful. See also the`request-timeout-interval` attribute. **Default:** 5 **Range**: 1 - 600

Attribute	Description
request-timeout-interval	This interval, when multiplied by the number of request retries, determines the maximum time (in seconds) to wait for the selected tunnel to establish before selecting to another tunnel from the list. **Default:** 1 **Range:** 1 - 30 ⚠ Modify the values for request-retries and request-timeout-interval attributes with caution, as incorrect value may lead to delay in tunnel establishment.
retransmit-interval	This value specifies the interval, in seconds, between retransmissions of a message. Each successive retransmission uses an exponential backoff interval, meaning the interval doubles after each retransmission. For example, if the first retransmission occurs after 1 second, the next will occur after 2 seconds, then 4 seconds, 8 seconds, 16 seconds, 32 seconds, and finally 64 seconds. After a maximum of 6 retransmissions, the interval reaches its maximum value. For a retransmit interval of 1, the total time can reach 64 seconds, and for an interval of 2, it can extend up to 128 seconds and so on. **Default:** 1 **Range**: 1 - 30
hide-authentication	When enabled, the L2TP proxy will hide the authentication response AVP if the authentication type is PAP, ensuring that the password is not transmitted in plain text. **Default:** false

Attribute	Description
pon-access-line-version	Adding additional PON attributes to the L2TP access line information. (For more information, see L2TP Access Line Information (RFC5515)) as defined in draft-lihawi-ancp-protocol-access-extension which can be optionally enabled using this configuration attribute. RFC and draft compliance are partial except as specified. The value DRAFT-LIHAWI-00 enables PON attributes based on the definition in draft-lihawi-ancp-protocol-access-extension-00 whereas DRAFT-LIHAWI-04 uses draft-lihawi-ancp-protocol-access-extension-04. **Default:**: DISABLED **Values:** DRAFT-LIHAWI-00, DRAFT-LIHAWI-04
connect-speed-update	Enable L2TP Connect-Speed-Update-Notification (CSUN) requests as defined in RFC5515. For more information, see Connect-Speed-Update-Notification (CSUN). CSUN is an L2TP control message sent by the LAC to the LNS to provide updates on transmit and receive connection speeds for one or more sessions. By default, this feature is disabled but can be enabled using this configuration. **Default:** false

Configuring L2TP over MPLS

L2TP over MPLS requires a dedicated L2TP service label which must be defined manually. First, you must assign a specific MPLS label for L2TP services. This is done in the L2TP profile configuration. Next, you must configure BGP to advertise this label. This involves configuring a routing policy that specifies how to handle the L2TP service label in BGP updates.

Following is an example command for L2TP configuration with L2TP service label.

```
set access l2tp-profile l2tp-default service-label 1234
```

Advertising this label through BGP must be configured manually as shown in the following example. The exact policy configuration depends on the actual network and existing policy concept.

In the following configuration, the policy named L2TP_MPLS contains rules for traffic related to L2TP over MPLS. The policy consists of multiple rules, each specifying conditions for matching traffic and actions to take.

In the 'Ordinal 1' match rule, it specifies the Type as 'ipv4-prefix', Value-Type as 'discrete', Match-Type as 'exact' and Value is set to '198.51.100.200/24'. This rule matches traffic that has an exact IPv4 prefix of 198.51.100.200/24.

In the Action Rule, it specifies Type as 'label', Operation as 'overwrite' and Value: label is set to '1337,bos:1'. If the traffic matches the IPv4 prefix, the action overwrites the MPLS label with '1337' and sets the BOS bit to '1'.

In the 'Ordinal 2', it defines the action rule. After applying the label overwrite action in 'Ordinal 1', this rule permits the traffic to continue through the network with the updated label.

```
supervisor@switch: cfg> show config policy
{
    "rtbrick-config:policy": {
      "statement": [
        {
          "name": "L2TP_MPLS",
          "ordinal": [
            {
              "ordinal": 1,
              "match": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "ipv4-prefix",
                    "value-type": "discrete",
                    "match-type": "exact",
                    "value": "198.51.100.200/24"
                  }
                ]
              },
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "label",
                    "operation": "overwrite",
                    "value": "label:1337,bos:1"
                  }
                ]
              }
            },
            {
              "ordinal": 2,
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "operation": "return-permit"
                  }
                ]
              }
            }
          ]
        }
      ]
    }
  }
```

In the following configuration, the instance name is set as ‘internet’ and the Address Family is specified as ‘ipv4 and unicast’. The export policy is set to L2TP_MPLS, indicating that the label defined in the L2TP_MPLS policy will be advertised for this instance.

```
supervisor@switch: cfg> show config instance internet
{
  "rtbrick-config:instance": {
    "name": "internet",
    "address-family": [
      {
        "afi": "ipv4",
        "safi": "unicast",
        "policy": {
          "export": "L2TP_MPLS"
        }
      }
    ]
  }
}
```

L2TP Tunnel Pool Configuration

The configuration of the Layer 2 Tunnel Protocol (L2TPv2) pool is optional for subscriber management. It is necessary only if you want to enable L2TP tunneling.

The L2TP pool configuration helps to efficiently organize and manage multiple LNS servers. It allows you to define these sets locally to ensure that L2TP tunnels are directed to the appropriate servers.

Configuring the L2TP Tunnel Pool

The following command and options allow you to configure an L2TP tunnel pool.

```
supervisor@switch: cfg> set access l2tp-pool
  <pool-name>             Name of the L2TP pool

supervisor@switch: cfg> set access l2tp-pool lns-servers
  <client-name>           L2TP client (LAC) name

supervisor@switch: cfg> set access l2tp-pool lns-servers BNG
  <server-name>           L2TP server (LNS) name

supervisor@switch: cfg> set access l2tp-pool lns-servers BNG LNS
  <cr>
  client-ipv4              L2TP client (LAC) IPv4
  preference               Preference [Range: <0-65535>]
  secret-encrypted-text    Shared secret in encrypted text
  secret-plain-text        Shared secret in plain text
  server-ipv4              L2PTP server (LNS) IPv4
  session-limit            Session limit [Range: <1-65535>]
```

The following example shows a local pool with two LNS severs.

This configuration sets an L2TP pool named 'lns-pool-example' with two LNS servers (LNS1 and LNS2). It defines how the device will manage L2TP tunnels. In this example, there are two different LNS server entries for two different LNS servers, LNS1 and LNS2. Both entries belong to the same pool named 'lns-pool-example'. Each entry in the l2tp-pool array defines the configuration for an individual LNS server.

The client-name is specified as BNG, which indicates that the client will initiate L2TP tunnels to the listed LNS servers. The server-name specifies the name of the LNS. Each entry has a unique server name, LNS1 for the first entry and LNS2 for the second.

The client-ipv4 specifies the IPv4 address of the L2TP client, that is BNG. In both entries, the LAC's IP address is '198.51.100.200'. The server-ipv4 specifies the IPv4 address of the LNS. The first entry has '198.51.100.219' for LNS1, and the second entry has '198.51.100.220' for LNS2.

The secret-encrypted-text parameter specifies the shared secret used for authenticating the L2TP tunnel between the LAC and LNS. The value is encrypted, and the secret is the same in both entries: '$21e4946e31b406de98b3077aef03ed5a7'.

The preference parameter sets the preference value for the LNS server. A lower value generally indicates a higher preference, but in this case, both entries have the same preference value of 1000. The session-limit parameter specifies the maximum number of sessions (L2TP tunnels) that can be established with the LNS server. Both entries have a session limit of 1000.

```
supervisor@switch: cfg> show config access
{
  "rtbrick-config:access": {
    "l2tp-pool": [
      {
        "pool-name": "lns-pool-example",
        "client-name": "BNG",
        "server-name": "LNS1",
        "client-ipv4": "198.51.100.200",
        "server-ipv4": "198.51.100.219",
        "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
        "preference": 1000,
        "session-limit": 1000
      },
      {
        "pool-name": "lns-pool-example",
        "client-name": "BNG",
        "server-name": "LNS2",
        "client-ipv4": "198.51.100.200",
        "server-ipv4": "198.51.100.220",
        "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
        "preference": 1000,
        "session-limit": 1000
      }
    ]
  }
}
```

Attribute	Description
client-name	Local L2TP tunnel client (LAC) hostname.
server-name	Remote L2TP tunnel server (LNS) hostname.
client-ipv4	Local L2TP tunnel client (LAC) IPv4 address.
server-ipv4	Remote L2TP tunnel server (LNS) IPv4 address.
secret-encrypted-text secret-plain-text	L2TP tunnel secret, which can be provided as plaintext or already encrypted text.
preference	L2TP tunnel preference where the lowest value has the highest priority. **Default:** 0 **Range:** 1 - 65535

Attribute	Description
session-limit	Tunnels with a session limit reached are not considered for further sessions. This limit has precedence over the default session limit specified in the l2tp-profile. **Default:** 64000 **Range:** 1 - 65535

User Profile Configuration

Subscribers are usually authenticated through remote servers like RADIUS. However, RBFS also provides the option to authenticate sessions by comparing them against user profiles that are locally defined within the system.

Configuring the User Profile

The following command and options are used to configure a user profile. A user profile also includes authentication details.

```
supervisor@switch: cfg> set access user-profile
  <user-name>            Username

supervisor@switch: cfg> set access user-profile user@rtbrick.com
  <cr>
  l2tp-pool-name            L2TP pool name
  password-encrypted-text   Secret/password in encrypted text
  password-plain-text       Secret/password in plain text
  tunnel-type               Tunnel type
```

The following configuration shows the user name specified as 'user@rtbrick.com'. The password-encrypted-text attribute enables you to set the user’s password in an encrypted format. The tunnel-type is set to PPPoE.

```
supervisor@switch: cfg> show config access user-profile user@rtbrick.com
{
  "rtbrick-config:user-profile": {
    "user-name": "user@rtbrick.com",
    "password-encrypted-text": "$243a1341f44f54888cdd385b9f40513f1",
    "tunnel-type": "PPPoE"
  }
}
```

Attribute	Description
user-name	Username of the subscriber.

Attribute	Description
password-encrypted-text password-plain-text	User password which can be provided as plaintext or already encrypted text.
tunnel-type	Subscriber tunnel type. **Default:** PPPoE **Values:** PPPoE, L2TP
l2tp-pool-name	Assign a local configured L2TP tunnel pool.

IP Address Pool Configuration

IP address pools are crucial for assigning IP addresses to subscribers, and configuring these pools is an essential for subscriber management.

The following image illustrates the IP address pool configuration and how they are associated with the other subscriber management tasks.

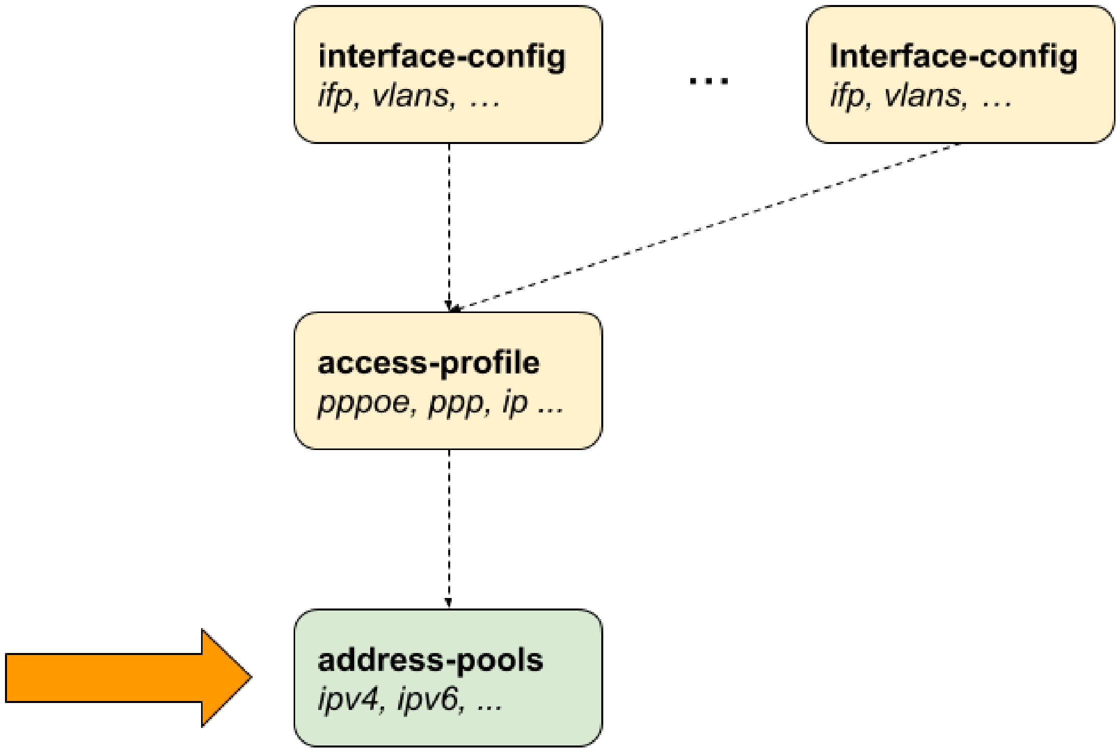


Figure 14. IP Address Pool Configuration

About RBFS IP Address Pool

RBFS allows you to manage address pools with more flexibility. It enables you to modify or remove pools without disrupting active subscriber sessions. If an address pool is deleted, subscribers who have already been assigned IP addresses from that pool can continue using those addresses until their session ends. The system does not immediately revoke the IP addresses assigned to active sessions.

Imagine a subscriber has been allocated an IP address from a specific address pool. If that pool is later deleted, the subscriber’s connection remains unaffected until their session ends.

Ensure that the deleted address pool is not reallocated elsewhere until all assigned addresses are no longer in use. This prevents IP address conflicts.

Duplicate Address Detection

RBFS provides an optional duplicate address detection feature that prevents a new session from using an IP address already in use by another active session. This is to avoid conflicts. However, when an address pool is moved to a different BNG, the duplicate address detection feature may not work as expected.

Migrating address pools across different BNGs could lead to address conflicts, if not managed carefully.

Configuring the Address Pool

The following command and options allows you to configure both IPv4 and IPv6 address pools for subscriber access.

```
   set access pool
  <pool-name>            Name of the address pool

supervisor@switch: cfg> set access pool ipv4-local
  <cr>
  ipv4-address           IPv4 address pool configuration
  ipv6-prefix            IPv6 prefix pool configuration
  next-pool-name         Name of the next address pool to be used if full
```

The following example shows typical IPv4 address and IPv6 prefix pools. It shows three pools: one for IPv4 addresses and two for IPv6 prefixes. Each pool is defined by a name, and a range of addresses or prefixes is specified using the 'low' and 'high' values. These pools are used for dynamically assigning IP addresses or prefixes to devices or subscribers, depending on the protocol (IPv4 or IPv6).

The IPv4 address pool, named 'ipv4-local', includes the address range from ‘198.51.100.76’ to ‘198.51.100.117’. This pool defines a range of IPv4 addresses from 198.51.100.76 to 198.51.100.117. These addresses can be dynamically assigned to devices or subscribers that require IPv4 connectivity.

The specified second pool is ‘ipv6-local’ that defines a range of IPv6 prefixes, from ‘2001:db8:0:79::/32’ to ‘2001:db8:0:139::/32’. These prefixes can be assigned to devices or subscribers that require IPv6 connectivity.

The IPv6 Prefix Delegation Pool is specified as ‘ipv6pd-local’. It defines the range of IPv6 from 2001:db8:0:1::/32 to ‘2001:db8:0:100::/32’. This pool is similar to the previous IPv6 pool, but is specifically meant for IPv6 Prefix Delegation.

```
supervisor@switch: cfg> show config access
{
  "rtbrick-config:access": {
    "pool": [
      {
        "pool-name": "ipv4-local",
        "ipv4-address": {
          "low": "198.51.100.76",
          "high": "198.51.100.117"
        }
      },
      {
        "pool-name": "ipv6-local",
        "ipv6-prefix": {
          "low": "2001:db8:0:79::/32",
          "high": "2001:db8:0:139::/32"
        }
      },
      {
        "pool-name": "ipv6pd-local",
        "ipv6-prefix": {
          "low": "2001:db8:0:1::/32",
          "high": "2001:db8:0:100::/32"
        }
      }
    ],
  }
}
```

Configuring IPv4 Address Pools

The following command and options allow you to configure an IPv4 address pool.

```
supervisor@switch: cfg> set access pool ipv4-local ipv4-address
  <cr>
  high                      Highest IPv4 address
  low                       Lowest IPv4 address
  subnet-mask               Subnet mask
```

Attribute	Description
high	Highest IPv4 address.
low	Lowest IPv4 address.
subnet-mask	subnet mask allocated to the subscriber.

Configuring IPv6 Prefix Pools

The following command and options allow you to configure an IPv6 prefix pools.

```
supervisor@switch: cfg> set access pool ipv6-local ipv6-prefix
  <cr>
  high                      Highest IPv6 prefix
  low                       Lowest IPv6 prefix
```

Attribute	Description
high	Highest IPv6 prefix.
low	Lowest IPv6 prefix.

IPv6 prefixes must be at least /64 or larger (/56, /48, …) or /128.

Configuring Linked Pools

RBFS enables pool linking. You can link multiple IP address pools so that when a pool gets exhausted, the next pool defined will starts address allocation. Pool linking allows the creation of discontinuous address pools connected together in a sequence, creating an extended pool. The pools are connected using the next-pool-name attribute.

A - B - C - D

```
supervisor@switch: cfg> set access pool pool-A next-pool pool-B
supervisor@switch: cfg> set access pool pool-B next-pool pool-C
supervisor@switch: cfg> set access pool pool-C next-pool pool-D
```

The actual address pool assigned to a subscriber is determined by the access configuration profile or RADIUS, starting with the specified pool in the chain. If that pool is full, the system requests the next pool in sequence, continuing until an available pool is found or the end of the chain is reached. To prevent looping, RBFS halts automatically if it detects that any pool in the chain has been accessed twice.

Additionally, the system can continuously cycle through all available pools in the chain, even if allocation starts from a pool in the middle. This ensures every pool in the sequence gets considered till finding an available address.

The following configuration commands link a set of IP address pools in a cycle order. The 'pool-A' is linked with the 'pool-B', 'pool-C' is set as a next pool after the 'pool-B'. 'Pool-D' is specified as next pool for 'pool-C'. 'Pool-A' is set as the next pool for 'pool-A'.

A - B - C - D - A

```
supervisor@switch: cfg> set access pool pool-A next-pool pool-B
supervisor@switch: cfg> set access pool pool-B next-pool pool-C
supervisor@switch: cfg> set access pool pool-C next-pool pool-D
supervisor@switch: cfg> set access pool pool-D next-pool pool-A
```

Control Plane Outbound Marking Configuration

The Control Plane Outbound Marking involves marking control plane packets as they exit the network device. This marking helps ensure that the packets are handled according to specific policies.

The process of marking all outbound control plane packets is established through the configuration settings within the forwarding-options, which are illustrated in the following example.

By employing these settings, network operators can ensure that control plane traffic is appropriately identified and treated according to specified criteria, contributing to an optimized and well-orchestrated network environment.

The following configuration shows the outbound control plane packets are marked with specific attributes, such as priority or type of service, making it easier to identify them within the network.

The configuration specifies how outbound control plane packets are marked based on their protocol under the forwarding-options. The class of service, is to set Control Plane QoS which ensures that the control traffic is marked and treated with priority. The outbound-marking defines how control plane traffic is marked. Protocol and Remark Types settings specify three protocols: L2TPv2, PPP, and RADIUS.

Each protocol is marked with different settings:

- The L2TPv2 packets are marked with a ToS codepoint of 64, which influences how routers handle these packets.
- The PPP packets are marked using a p-bit (priority bit) of 6, which indicates priority.
- Similar to L2TPv2, RADIUS packets are marked with a ToS codepoint of 64 for priority.

```
supervisor@switch: cfg> show config forwarding-options
{
    "rtbrick-config:forwarding-options": {
      "class-of-service": {
        "control-plane-qos": {
          "outbound-marking": {
            "protocol": [
              {
                "protocol": "l2tpv2",
                "remark-type": "tos",
                "codepoint": 64
              },
              {
                "protocol": "ppp",
                "remark-type": "p-bit",
                "codepoint": 6
              },
              {
                "protocol": "radius",
                "remark-type": "tos",
                "codepoint": 64
              }
            ]
          }
        }
      }
    }
  }
```

Configuration Examples

PPPoE

The following example shows a PPPoE configuration for VLAN mode 1:1 with IPv4 and IPv6 enabled, authenticated via RADIUS.

```
{
  "ietf-restconf:data": {
    "rtbrick-config:access": {
      "aaa-profile": [
        {
          "profile-name": "aaa-radius",
          "session-timeout": 0,
          "idle-timeout": 0,
          "aaa-radius-profile": "radius-default",
          "authentication": {
            "order": "RADIUS"
          },
          "accounting": {
            "order": "RADIUS",
          }
        }
      ],
      "radius-profile": [
        {
          "profile-name": "radius-default",
          "nas-identifier": "BNG",
          "nas-port-type": "Ethernet",
          "authentication": {
            "radius-server-profile-name": [
              "radius-server-1",
              "radius-server-2"
              ]
          },
          "accounting": {
            "radius-server-profile-name": [
              "radius-server-1",
              "radius-server-2"
              ],
            "stop-on-reject": "true",
            "stop-on-failure": "true",
            "accounting-on-off": "true",
            "accounting-on-wait": "true",
            "accounting-backup": "true",
            "accounting-backup-max": 86400
          }
        }
      ],
      "radius-server": [
        {
          "server-name": "radius-server-1",
          "address": "198.51.100.101",
          "source-address": "198.51.100.200",
          "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
          "authentication": {
            "enable": "true"
          },
          "accounting": {
            "enable": "true"
          },
          "coa": {
            "enable": "true"
          }
        },
        {
          "server-name": "radius-server-2",
          "address": "198.51.100.102",
          "source-address": "198.51.100.200",
          "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
          "authentication": {
            "enable": "true"
          },
          "accounting": {
            "enable": "true"
          },
          "coa": {
            "enable": "true"
          }
        }
      ],
      "access-profile": [
        {
          "profile-name": "pppoe-dual",
          "protocol": {
            "pppoe": {
              "enable": "true",
              "session-protection": {
                "enable": "true"
              },
              "vlan-priority": 6
            },
            "ppp": {
              "lcp": {
                "authentication-protocol": "PAP_CHAP",
                "echo-interval": 30,
                "echo-max-retransmit": 3,
                "echo-enable": "true"
              },
              "ipcp": {
                "enable": "true",
                "source-ifl": "lo-0/0/0/1"
              },
              "ip6cp": {
                "enable": "true"
              }
            },
            "ra": {
              "enable": "true",
              "interval": 60
            },
            "dhcpv6": {
              "enable": "true"
            },
            "l2tp": {
              "tunnel-profile": "l2tp-default"
            }
          },
          "address-family": {
            "ipv4": {
              "enable": "true",
              "primary-dns": "198.51.100.103",
              "secondary-dns": "198.51.100.104",
              "instance": "default"
            },
            "ipv6": {
              "enable": "true",
              "primary-dns": "2001:db8:0:100::",
              "secondary-dns": "2001:db8:0:104::",
              "instance": "default"
            }
          }
        }
      ],
      "interface": {
        "double-tagged": [
          {
            "interface-name": "ifl-0/0/1",
            "outer-vlan-min": 1,
            "outer-vlan-max": 4094,
            "inner-vlan-min": 7,
            "inner-vlan-max": 7,
            "access-type": "PPPoE",
            "access-profile-name": "pppoe-dual",
            "aaa-profile-name": "aaa-radius"
          }
        ]
      },
      "l2tp-profile": [
        {
          "profile-name": "l2tp-default",
          "session-limit": 4000,
          "client-name": "BNG",
          "client-ipv4": "198.51.100.200",
          "hide-authentication": true
        }
      ]
    },
    "rtbrick-config:interface": [
      {
        "name": "ifl-0/0/1",
        "description": "Access",
        "host-if": "eth0"
      },
      {
        "name": "ifl-0/0/2",
        "description": "Core",
        "host-if": "eth1",
        "unit": [
          {
            "unit-id": 1,
            "address": {
              "ipv4": [
                {
                  "prefix4": "198.51.100.33/24"
                }
              ],
              "ipv6": [
                {
                  "prefix6": "2001:db8:0:32::/32"
                }
              ]
            }
          }
        ]
      },
      {
        "name": "lo-0/0/0",
        "unit": [
          {
            "unit-id": 1,
            "address": {
              "ipv4": [
                {
                  "prefix4": "198.51.100.200/24"
                }
              ]
            }
          }
        ]
      }
    ]
  }
}
```

IPoE

The following example shows an IPoE configuration for VLAN mode 1:1 with IPv4 and IPv6 enabled, authenticated via RADIUS.

```
{
  "ietf-restconf:data": {
    "rtbrick-config:access": {
      "aaa-profile": [
        {
          "profile-name": "aaa-radius",
          "session-timeout": 0,
          "idle-timeout": 0,
          "aaa-radius-profile": "radius-default",
          "authentication": {
            "order": "RADIUS"
          },
          "accounting": {
            "order": "RADIUS",
          }
        }
      ],
      "radius-profile": [
        {
          "profile-name": "radius-default",
          "nas-identifier": "BNG",
          "nas-port-type": "Ethernet",
          "authentication": {
            "radius-server-profile-name": [
              "radius-server-1",
              "radius-server-2"
              ]
          },
          "accounting": {
```

```
          "radius-server-profile-name": [
            "radius-server-1",
            "radius-server-2"
            ],
          "stop-on-reject": "true",
          "stop-on-failure": "true",
          "accounting-on-off": "true",
          "accounting-on-wait": "true",
          "accounting-backup": "true",
          "accounting-backup-max": 86400
        }
      }
    ],
    "radius-server": [
      {
        "server-name": "radius-server-1",
        "address": "198.51.100.101",
        "source-address": "198.51.100.200",
        "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
        "authentication": {
          "enable": "true"
        },
        "accounting": {
          "enable": "true"
        },
        "coa": {
          "enable": "true"
        }
      },
      {
        "server-name": "radius-server-2",
        "address": "198.51.100.102",
        "source-address": "198.51.100.200",
        "secret-encrypted-text": "$21e4946e31b406de98b3077aef03ed5a7",
        "authentication": {
          "enable": "true"
        },
        "accounting": {
          "enable": "true"
        },
        "coa": {
          "enable": "true"
        }
      }
    ],
    "access-profile": [
      {
        "profile-name": "ipoe-dual",
        "protocol": {
          "dhcp": {
            "enable": "true",
            "mode": "server"
          },
          "dhcpv6": {
            "enable": "true",
            "mode": "server"
          },
        },
        "address-family": {
          "ipv4": {
            "enable": "true",
            "pool-name":"ipoe",
            "primary-dns": "198.51.100.103",
            "secondary-dns": "198.51.100.104",
            "instance": "default"
          },
          "ipv6": {
            "enable": "true",
            "pool-name":"ipoe-ia-na",
            "prefix-delegation-pool-name":"ipoe-ia-pd",
            "primary-dns": "2001:db8:0:100::",
            "secondary-dns": "2001:db8:0:104::",
            "instance": "default"
          }
        }
      }
    ],
    "interface": {
      "double-tagged": [
        {
          "interface-name": "ifl-0/0/1",
          "outer-vlan-min": 1,
          "outer-vlan-max": 4094,
          "inner-vlan-min": 7,
          "inner-vlan-max": 7,
          "access-type": "IPoE",
          "access-profile-name": "ipoe-dual",
          "aaa-profile-name": "aaa-radius",
          "gateway-ifl": "lo-0/0/0/1"
        }
      ]
    },
    "pool": [
      {
        "pool-name": "ipoe",
        "ipv4-address": {
          "low": "10.0.0.1",
          "high": "10.0.255.255"
        }
      },
      {
        "pool-name": "ipoe-ia-na",
        "ipv6-prefix": {
          "low": "fc66::1/128",
          "high": "fc66::ffff/128"
        }
      },
      {
        "pool-name": "ipoe-ia-pd",
        "ipv6-prefix": {
          "low": "fc66:0:100::/56",
          "high": "fc66:0:1ff:ff00::/56"
        }
      }
    ],
  },
  "rtbrick-config:interface": [
    {
      "name": "ifl-0/0/1",
      "description": "Access",
      "host-if": "eth0"
    },
        {
          "name": "ifl-0/0/2",
          "description": "Core",
          "host-if": "eth1",
          "unit": [
            {
              "unit-id": 1,
              "address": {
                "ipv4": [
                  {
                    "prefix4": "198.51.100.33/24"
                  }
                ],
                "ipv6": [
                  {
                    "prefix6": "2001:db8:0:6423::/32"
                  }
                ]
              }
            }
          ]
        },
        {
          "name": "lo-0/0/0",
          "unit": [
            {
              "unit-id": 1,
              "address": {
                "ipv4": [
                  {
                    "prefix4": "198.51.100.200/24"
                  }
                ]
              }
            }
          ]
        }
      ]
    }
  }
```

5.1.3. Operations

Subscriber Management

The following commands are served by subscriber daemon and are applicable for all kinds of subscribers like PPPoE, L2TP or IPoE.

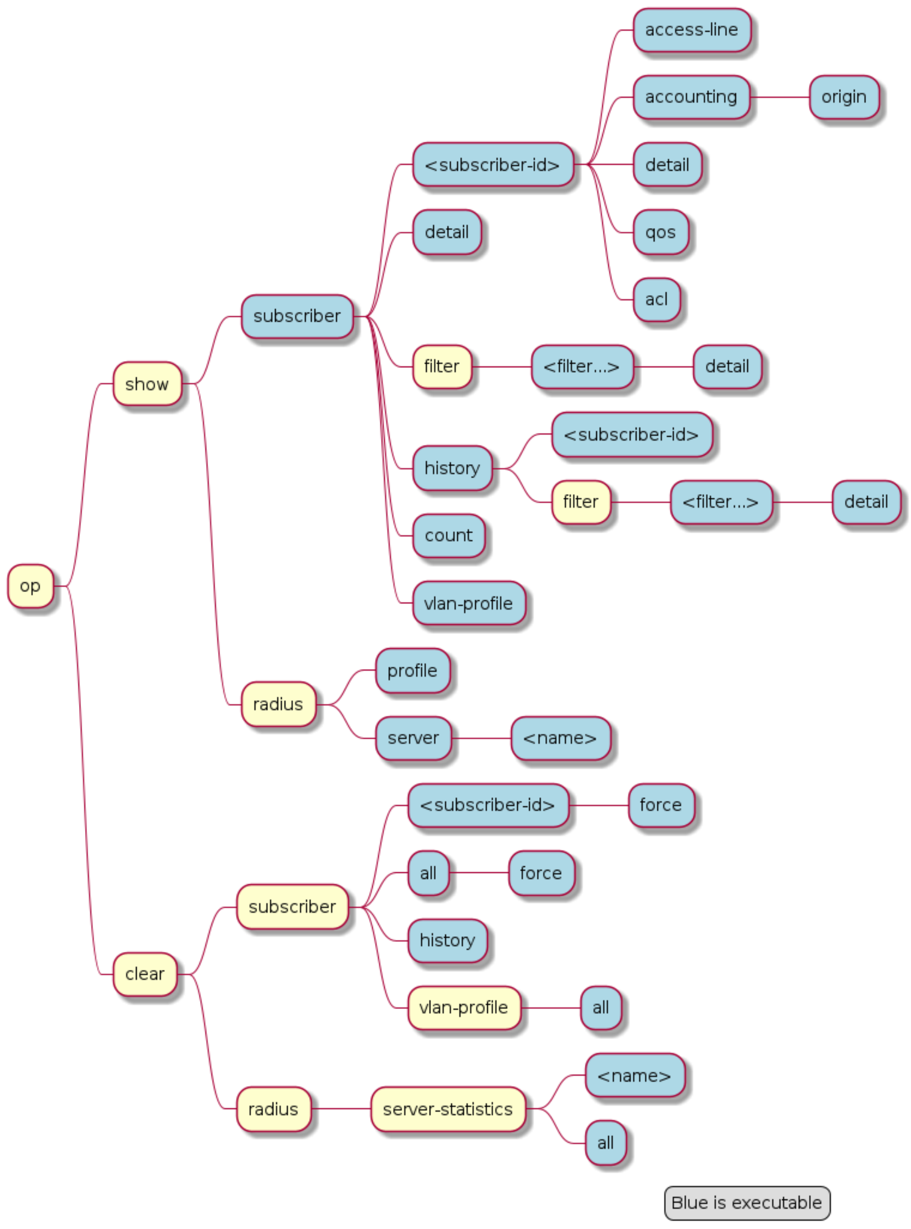


Figure 15. Subscriber Management Operational Commands

Subscribers

The term subscriber describes an access user or session from a higher level decoupled from underlying protocols like PPPoE or IPoE. Subscribers in RBFS can be managed locally or remote via RADIUS. Each subscriber is uniquely identified by a 64bit number called subscriber-id.

Subscriber States

A good starting point for troubleshooting subscriber services is to verify the status of the subscriber sessions. The state ESTABLISHED means that the session is fully operational.

```
supervisor@leaf1: op> show subscriber
Subscriber-Id          Interface        VLAN      Type   State
72339069014638600      ifp-0/0/1        1:1       PPPoE  ESTABLISHED
72339069014638601      ifp-0/0/1        1:2       PPPoE  ESTABLISHED
72339069014638602      ifp-0/0/1        1:3       PPPoE  ESTABLISHED
72339069014638603      ifp-0/0/3        2000:7    L2TP   ESTABLISHED
```

Alternative use show subscriber detail which shows further details like username, Agent-Remote-Id (aka Line-Id) or Agent-Circuit-Id if screen width is large enough to print all those information.

The meaning of the subscriber state is shown in the following table and diagram.

State	Description
INIT	Initial subscriber state.
AUTHENTICATING	Authenticate the subscriber using the configured method.
ADDRESS ALLOCATION	Allocate (RADIUS or pool) and validate (DAD) addresses.
TUNNEL SETUP	Setup tunnel resources (L2TP or L2X).
IFL SETUP	Create subscriber IFL with corresponding QoS resources.
FULL	Wait for subscriber to be in forwarding state. Inform underlying protocols (PPPoED or IPoED) to continue with session setup.
ACCOUNTING	Start subscriber accounting and wait for response.
ESTABLISHED	The subscriber becomes ESTABLISHED after response to RADIUS Accounting-Request-Start if RADIUS accounting is enabled otherwise immediately after FULL.
TERMINATING	The subscriber remains in this state until all resources are freed and accounting stopped. This means that subscriber remain in this state until response to RADIUS Accounting-Request-Stop if RADIUS accounting is enabled.

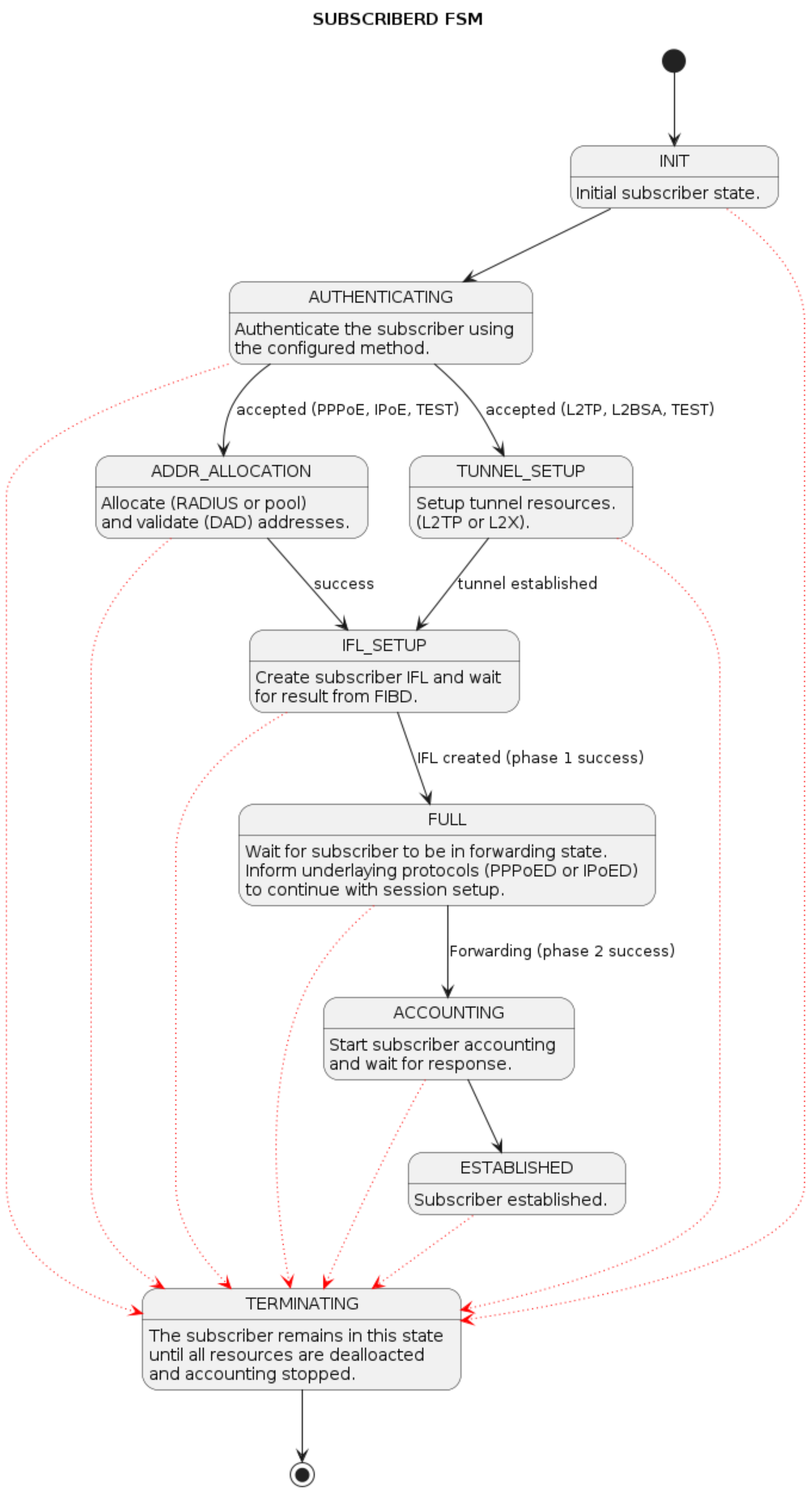

SUBSCRIBERD FSM
INIT
Initial subscriber state.
AUTHENTICATING
Authenticate the subscriber using the configured method.
accepted (PPPoE, IPoE, TEST)
accepted (L2TP, L2BSA, TEST)
ADDR_ALLOCATION
Allocate (RADIUS or pool) and validate (DAD) addresses.
TUNNEL_SETUP
Setup tunnel resources. (L2TP or L2X).
success
tunnel established
IFL_SETUP
Create subscriber IFL and wait for result from FIBD.
IFL created (phase 1 success)
FULL
Wait for subscriber to be in forwarding state. Inform underlaying protocols (PPPoED or IPoED) to continue with session setup.
Forwarding (phase 2 success)
ACCOUNTING
Start subscriber accounting and wait for response.
ESTABLISHED
Subscriber established.
TERMINATING
The subscriber remains in this state until all resources are dealloacted and accounting stopped.

Figure 16. Subscriber States

For each subscriber a set of commands is available showing detailed information.

```
supervisor@leaf1: op> show subscriber 72339069014638594
  <cr>
  access-line           Subscriber access line information
  accounting            Subscriber accounting information
  acl                   Subscriber ACL information (filter)
  detail                Detailed subscriber information
  qos                   Subscriber QoS information

user@switch: op> show subscriber 72339069014638594 detail
Subscriber-Id: 72339069014638594
    Type: PPPoE
    State: ESTABLISHED
    Created: Fri Sep 18 20:50:02 GMT +0000 2020
    Interface: ifl-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    Client MAC: fe:08:e8:ea:1d:32
    Server MAC: 7a:52:4a:01:00:01
    IFL: ppp-0/0/1/72339069014638594
    Username: 1122334455#123456789#0001@t-online.de
    Agent-Remote-Id: DEU.DTAG.1337
    Agent-Circuit-Id: 0.0.0.0/0.0.0.0 eth 1337
    Access-Profile: access-profile1
    AAA-Profile: aaa-profile1
    Session-Timeout: 30000
    Idle-Timeout: 120
    IPv4:
        Instance: default
        Address: 198.51.100.116/255.255.255.255
        Address Active: True
        Primary DNS: 198.51.100.213
        Secondary DNS: 198.51.100.54
    IPv6:
        Instance: default
        RA Prefix: 2001:db8:0:400::/32
        RA Prefix Active: True
        Delegated Prefix (DHCPv6): 2001:db8:0:269::/56
        Delegated Prefix Active: False
        Primary DNS: 2001:db8:0:92::
        Secondary DNS: 2001:db8:0:174::
    Accounting:
        Session-Id: 72339069014638594:1600462202
        Start-Time: 2020-09-18T20:50:02.738306+0000
        Interims Interval: 30 seconds
```

Subscriber Termination Codes

The following command shows the reasons why subscribers are terminated for the last 24 hours and up to 4000 subscribers.

```
supervisor@leaf1: op> show subscriber history
Subscriber-Id          Timestamp                             Terminate Code
72339069014638594      Fri Oct 16 20:17:33 GMT +0000 2020    Accounting-Request-On
Wait
72339069014638595      Fri Oct 16 20:32:19 GMT +0000 2020    PPPoE LCP Terminate
Request Received
```

Subscriber Count

To view a summary of PPPoE, L2TP, IPoE, and L2BSA subscribers in the Setup, Established, and Terminating state, use the "show subscriber count" command. This command provides information per interface and a total summary based on subscriber type.

```
supervisor@leaf1: op> show subscriber count
                       Total        Setup        Established  Terminating
Summary                18000        0            18000        0
  PPPoE                18000        0            18000        0
  L2TP                 0            0            0            0
  IPoE                 0            0            0            0
  L2BSA                0            0            0            0
ifp-0/1/30             6000         0            6000         0
  PPPoE                6000         0            6000         0
  L2TP                 0            0            0            0
  IPoE                 0            0            0            0
  L2BSA                0            0            0            0
ifp-0/1/32             6000         0            6000         0
  PPPoE                6000         0            6000         0
  L2TP                 0            0            0            0
  IPoE                 0            0            0            0
  L2BSA                0            0            0            0
ifp-0/1/33             6000         0            6000         0
  PPPoE                6000         0            6000         0
  L2TP                 0            0            0            0
  IPoE                 0            0            0            0
  L2BSA                0            0            0            0
supervisor@leaf1: op>
```

RADIUS

RADIUS Profile

The following command shows the status of all RADIUS profiles.

```
supervisor@leaf1: op> show radius profile
RADIUS Profile: radius-default
    NAS-Identifier: BNG
    NAS-Port-Type: Ethernet
    Authentication:
        Algorithm: ROUND-ROBIN
        Server:
            radius-server-1
            radius-server-2
    Accounting:
        State: UP
        Stop on Reject: True
        Stop on Failure: True
        Backup: True
        Algorithm: ROUND-ROBIN
        Server:
            radius-server-1
            radius-server-2
```

This meaning of the accounting state is explained in the table below.

Code	State	Description
0x00	DISABLED	Change profile accounting state from DISABLED to ACTIVE if at least one server referenced is found with accounting enabled.
0x01	ACTIVE	Server referenced by RADIUS profile but no response received
0x02	STARTING	Send accounting-on and wait for response.
0x05	UP	Change profile accounting state to UP if at least one referenced accounting server is UP.

The profile state becomes immediately ACTIVE if at least one of the referenced accounting servers can be found in RADIUS server table with accounting enabled. Otherwise the profile keeps DISABLED.

If RADIUS Accounting-On is enabled, the profile state becomes STARTING before UP. It is not permitted to send any accounting request start, interim or stop related to a profile in this state. It is also not permitted to send authentication requests if **accounting-on-wait** is configured in addition. The state becomes UP if at least one server in the accounting server list is in a state UP or higher (UNREACHABLE, DOWN, TESTING, DEAD).

A new profile added which references existing used RADIUS servers must not trigger a RADIUS Accounting-On request if at least one of the referenced servers is in a state of UP or higher.

RADIUS Server

The following command shows the status of all RADIUS servers.

```
supervisor@leaf1: op> show radius server
RADIUS Server           Address          Authentication State Accounting State
radius-server-1         198.51.100.64    ACTIVE               UP
radius-server-2         198.51.100.163   ACTIVE               ACTIVE
radius-server-3         198.51.100.104   ACTIVE               ACTIVE
```

This meaning of those states is explained in the table and diagram below.

Code	State	Description
0x00	DISABLED	RADIUS authentication (authentication state) or accounting (accounting state) is disabled or server not referenced by profile.
0x01	ACTIVE	Server referenced by RADIUS profile but no valid response received.
0x02	STARTING	This state is valid for accounting (accounting state) only during accounting-on is sending (wait for accounting-on response).
0x03	STOPPING	This state is valid for accounting (accounting state) only during accounting-off is sending (wait for accounting-off response).
0x04	FAILED	This state is valid for accounting (accounting state) only if accounting-on/off timeout occurs.
0x05	UP	Valid RADIUS response received
0x06	UNREACHABLE	No response received/timeout but server is still usable.
0x07	DOWN	Server is down but can be selected.
0x08	TESTING	Send a request to test if server is back again. The server will not be selected for another request in this state (use a single request to check if server is back again).
0x09	DEAD	Server is down and should not be selected.

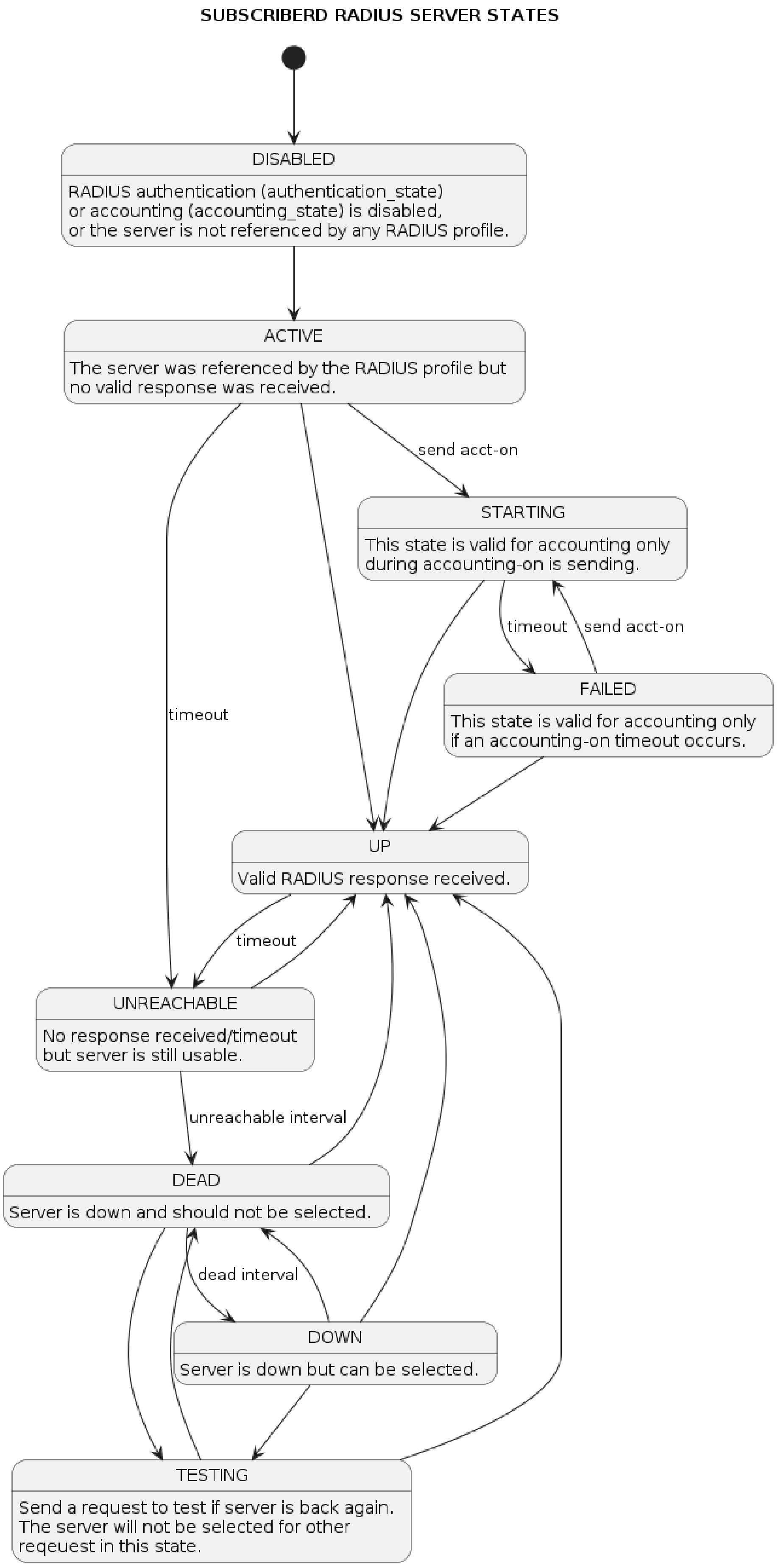

SUBSCRIBERD RADIUS SERVER STATES
DISABLED
RADIUS authentication (authentication_state)
or accounting (accounting_state) is disabled,
or the server is not referenced by any RADIUS profile.
ACTIVE
The server was referenced by the RADIUS profile but
no valid response was received.
send acct-on
STARTING
This state is valid for accounting only
during accounting-on is sending.
timeout
send acct-on
FAILED
This state is valid for accounting only
if an accounting-on timeout occurs.
timeout
UP
Valid RADIUS response received.
timeout
UNREACHABLE
No response received/timeout
but server is still usable.
unreachable interval
DEAD
Server is down and should not be selected.
dead interval
DOWN
Server is down but can be selected.
TESTING
Send a request to test if server is back again.
The server will not be selected for other
reqeuest in this state.

Figure 17. RADIUS Server States

For each server dedicated detailed information are displayed with the following commands.

```
supervisor@leaf1: op> show radius server radius-server-1
RADIUS Server: radius-server-1
    Address: 198.51.100.64
    Source: 198.51.100.200
    Rate: 600 PPS
    Rate Tokens: 600
    Dropped: 0
    Authentication:
        State: ACTIVE
        State Changed: Fri Oct 16 20:17:27 GMT +0000 2020
        Port: 1812
        Retry: 3
        Timeout: 5
        Outstanding: 100
        Statistics:
            Request Sent: 0
            Request Retry: 0
            Request Timeout: 0
            Accept Received: 0
            Reject Received: 0
            Dropped: 0
    Accounting:
        State: UP
        State Changed: Fri Oct 16 20:18:27 GMT +0000 2020
        Port: 1813
        Retry: 10
        Timeout: 30
        Outstanding: 100
        Statistics:
            Request Sent: 1
            Request Retry: 2
            Request Timeout: 0
            Response Received: 1
            Dropped: 0
    CoA:
        Port: 3799
        Statistics:
            Request Received: 0
            Dropped: 0
```

PPPoE

The following commands are applicable for PPPoE sessions only.

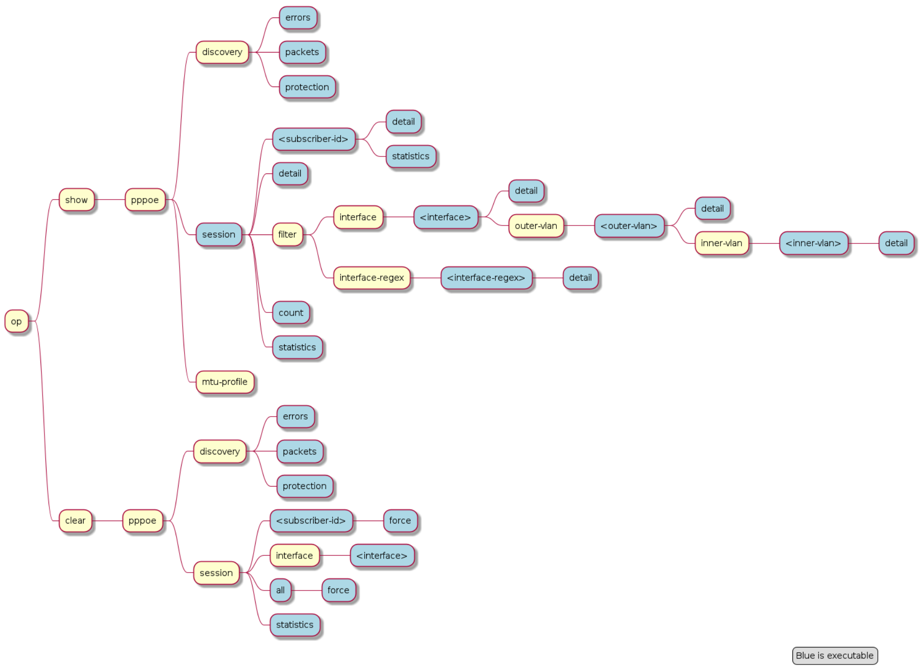


Figure 18. PPPoE Operational Commands

For PPPoE sessions the state should be ESTABLISHED if local terminated or TUNNELLED for L2TPv2 tunnelled sessions.

```
supervisor@rtbrick: op> show pppoe session
Subscriber-Id        Interface        VLAN      MAC               State
72339069014638604    ifp-0/0/1        1:1       00:04:0e:00:00:01 ESTABLISHED
72339069014638601    ifp-0/0/1        1:2       00:04:0e:00:00:02 ESTABLISHED
72339069014638602    ifp-0/0/1        1:3       00:04:0e:00:00:03 ESTABLISHED
72339069014638603    ifp-0/0/3        2000:7    52:54:00:57:c8:29 TUNNELLED
```

Alternative use show pppoe session detail which shows further details like username, Agent-Remote-Id (aka Line-Id) or Agent-Circuit-Id if screen width is large enough to print all those information.

State	Description
LINKING	PPP LCP setup.
AUTHENTICATING	PPP authentication (PAP or CHAP).
NETWORKING	PPP IPCP (IPv4) and IP6CP (IPv6) setup.
ESTABLISHED	The PPPoE session becomes established if at least one NCP (IPCP or IP6CP) is established (state OPEN).
TUNNELLED	This state indicates that a PPPoE session is tunnelled via L2TPv2.
TERMINATING	PPP session teardown.

State	Description
TERMINATED	PPPoE session terminated.

If PPPoE session remain in state TERMINATED, the subscriber state should be checked. Typically this happens if RADIUS Accounting-Request-Stop is still pending.

Further details per PPPoE session can be shown with the following commands.

```
supervisor@rtbrick: op> show pppoe session 72339069014638648
  <cr>
  detail                Detailed session information
  statistics            Protocol statistics
```

The detail command shows the states of the session and all sub-protocols with extensive information and negotiated parameters.

```
user@switch: op> show pppoe session 72339069014638648 detail
Subscriber-Id: 72339069014638648
    State: ESTABLISHED
    Uptime: Tue Nov 17 11:46:43 GMT +0000 2020 (0:00:21.979775)
    Interface: ifp-0/0/3
    Outer VLAN: 10
    Inner VLAN: 7
    Client MAC: 52:54:00:57:c8:29
    Server MAC: 7a:52:4a:c0:00:03
    Session-Id: 55
    Host-Unique: 00000001
    Agent-Remote-Id: DEU.RTBRICK.1
    Agent-Circuit-Id: 0.0.0.0/0.0.0.0 eth 1
    Access-Profile: pppoe-dual
    AAA-Profile: aaa-default
    PPP LCP:
        State: OPENED
        Negotiated Protocols: CHAP, IPCP, IP6CP
        Negotiated Parameters: MRU, AUTH, MAGIC
        Magic Number: 1079931229 Peer: 3432759752
        MRU: 1492 Peer: 1492
        MTU: 1492 Profile: __default_pppoe__
        Echo Interval: 30 seconds
    CHAP Authentication:
        State: COMPLETED
        Username: user1@rtbrick.com
    PPP IPCP:
        State: OPENED
        Instance: default
        IP Address: 198.51.100.200 Peer: 198.51.100.72
        Primary DNS: 198.51.100.88
        Secondary DNS: 198.51.100.54
    PPP IP6CP:
        State: OPENED
        Instance: default
        Interface Identifier: c5f6:1dbd:8cc1:bea9
        Peer Interface Identifier: 5054:00ff:fe57:c829
    IPv6:
        RA Interval: 60 seconds
        RA Prefix: 2001:db8:0:246::/32
        Delegated Prefix (DHCPv6): 2001:db8:0:9::/32 Assigned: True
        Primary DNS: 2001:db8:0:114::
        Secondary DNS: 2001:db8:0:115::
    Control Traffic Statistics:
        Ingress: 15 packets 1059 bytes
        Egress: 16 packets 1475 bytes
```

Session statistics are available global and per session.

```
supervisor@rtbrick: op> show pppoe session statistics
supervisor@rtbrick: op> show pppoe session 72339069014638601 statistics
```

The PPPoE discovery statistics are helpful if session setup fails in initial PPPoE tunnel setup before actual PPP negotiation is starting.

```
supervisor@rtbrick: op> show pppoe discovery packets
Packet            Received          Sent
PADI              17                0
PADO              0                 17
PADR              17                0
PADS              0                 17
PADT              1                 13

supervisor@rtbrick: op> show pppoe discovery errors
PADI Drop No Config               : 0
PADI Drop Session Protection      : 0
PADI Drop Session Limit           : 0
PADI Drop Dup Session             : 0
PADI Drop Interface Down          : 0
PADR Drop No Config               : 0
PADR Drop Wrong MAC               : 0
PADR Drop Interface Down          : 0
PADR Drop Session Limit           : 0
PADR Drop Session Protection      : 0
PADR Drop Bad Cookie              : 0
PADR Drop Bad Session             : 0
PADR Drop Dup Session             : 0
PADR Drop No mapping Id           : 0
PADT Drop No Session              : 0
PADT Drop Wrong MAC               : 0
PADX Interface Get Failure        : 0
```

If PPPoE session protection is enabled in access configuration profile, short lived or failed sessions will be logged in the PPPoE session protection table (local.pppoe.session.protection).

Every session not established for at least 60 seconds per default is considered as failed or short lived session. This will block new sessions on this IFP and VLAN's for one second per default which increase exponential with any further failed session until the max time of per default 300 seconds is reached. The interval is reset after 900 seconds without failed sessions.

The PPPoE session protection table include also last subscriber-id and terminate code which indicates the reason for session failures.

```
supervisor@rtbrick: op> show pppoe discovery protection
Interface        VLAN      Status  Attempts   Last Terminate Code
ifp-0/0/1        1:1       OK      1          PPPoE LCP Terminate Request Received
ifp-0/0/1        1:2       OK      1          PPPoE LCP Terminate Request Received
ifp-0/0/1        1:3       OK      1          PPPoE LCP Terminate Request Received
```

If status OK indicates that new session are accepted where BLOCKED means that sessions will be rejected.

L2TP

The following commands are applicable for L2TP only.

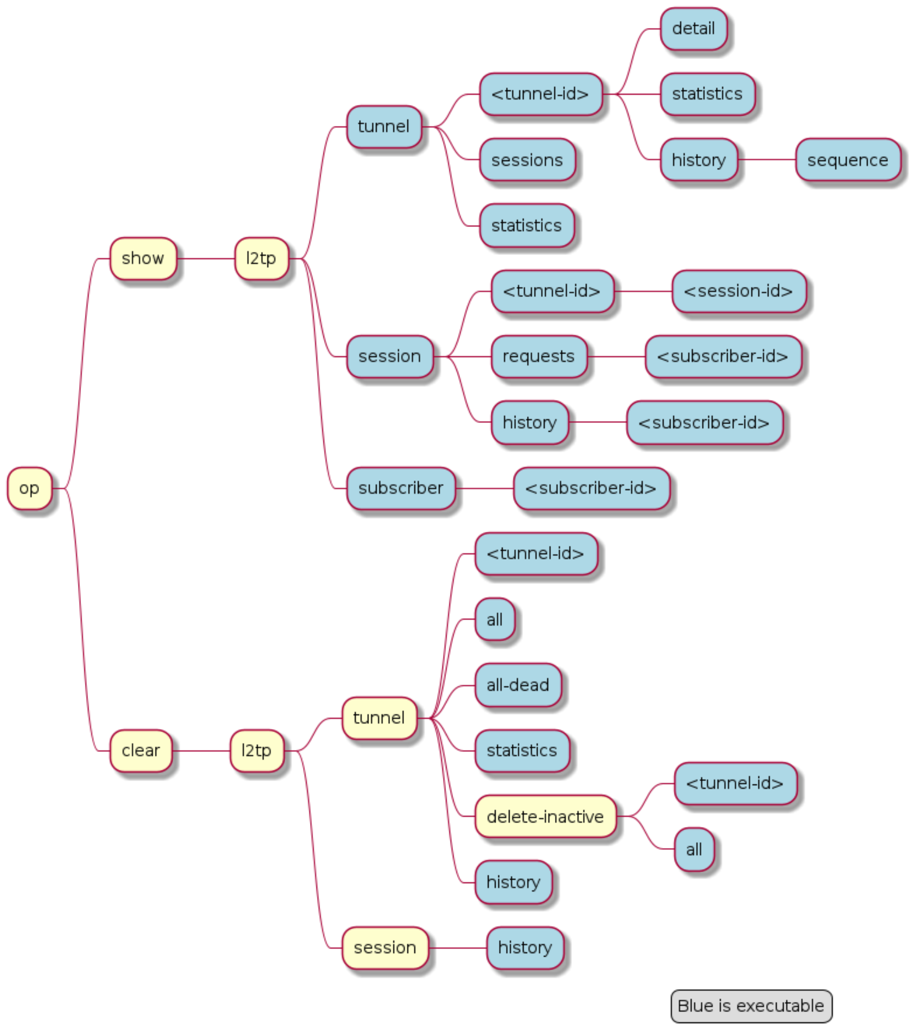


Figure 19. L2TP Operational Commands

For L2TPv2 tunnelled PPPoE sessions the global unique subscriber-id can be used to get information about the L2TP session.

```
supervisor@rtbrick: op> show l2tp subscriber 72339069014638621
Subscriber-Id: 72339069014638621
    State: ESTABLISHED
    Local TID: 45880
    Local SID: 39503
    Peer TID: 1
    Peer SID: 1
    Call Serial Number: 10
    TX Speed: 10007000 bps
    RX Speed: 1007000 bps
    CSUN: disabled
```

The following command gives a good overview over the corresponding tunnels.

```
supervisor@leaf1: op> show l2tp tunnel sessions
Role Local TID Peer TID State       Preference Sessions Established Peer Name
LAC       2022        1 ESTABLISHED      10000        1           1 LNS3
LAC       3274         1 ESTABLISHED       10000        1           1 LNS8
LAC      14690         1 ESTABLISHED       10000        1           1 LNS6
LAC      29489         1 ESTABLISHED       10000        1           1 LNS9
LAC      33323         1 ESTABLISHED       10000        1           1 LNS4
LAC      35657         1 ESTABLISHED       10000        1           1 LNS10
LAC      37975         1 ESTABLISHED       10000        1           1 LNS1
LAC      45880         1 ESTABLISHED       10000        1           1 LNS7
LAC      46559         1 ESTABLISHED       10000        1           1 LNS2
LAC      58154         1 ESTABLISHED       10000        1           1 LNS5
```

Detailed information per tunnel are available via show l2tp tunnel <TID> detail.

L2TP tunnel statistics are available global and per tunnel.

```
supervisor@leaf1: op> show l2tp tunnel statistics
supervisor@leaf1: op> show l2tp tunnel 37975 statistics
```

L2TP Result and Disconnect Codes

The received result (RFC2661) and disconnect (RFC3145) code and message from CDN and StopCCN will be stored similar to the subscriber terminate history table for 24 hours and up to 1000 records.

```
supervisor@leaf1: op> show l2tp tunnel history
Sequence Local TID Peer TID Timestamp                          Terminate Code
       1     34209        0 Wed Jul 28 13:02:35 GMT +0000 2021   Admin Request
       2     39860        1 Wed Jul 28 13:02:35 GMT +0000 2021   Admin Request
       3     39860        2 Wed Jul 28 13:02:54 GMT +0000 2021   Admin Request
       4     39860        3 Wed Jul 28 13:04:29 GMT +0000 2021   StopCCN Received
(Requester is being shut down)
       5     39860        1 Wed Jul 28 13:06:19 GMT +0000 2021   StopCCN Received
(Requester is being shut down)

supervisor@leaf1: op> show l2tp tunnel history 4
Local TID: 39860 Peer TID: 3
    Terminate Code: StopCCN Received
    Timestamp: Wed Jul 28 13:04:29 GMT +0000 2021
    Local Address: 198.51.100.102
    Peer Address: 198.51.100.133
    Peer Name: LNS1
    Tunnel-Client-Auth-ID: BNG
    Tunnel-Server-Auth-ID: LNS1
    Result Code: Requester is being shut down

supervisor@leaf1: op> show l2tp session history
Subscriber-Id          Local TID Local SID Terminate Code
72339069014638614          39860      5597 Clear Session
72339069014638615          39860      5208 Clear Session
72339069014638623          39860     29626 Clear Session
72339069014638624          39860     42480 L2TP Tunnel Down
72339069014638625          39860     34417 L2TP Tunnel Down
72339069014638626          39860     20229 L2TP Tunnel Down
```

The show subscriber history <subscriber-id> command will also return L2TP details if found for the corresponding subscriber.

```
supervisor@leaf1: op> show subscriber history 72339069014638703
Subscriber-Id: 72339069014638703
    Terminate Code: L2TP CDN Request
    Timestamp: Wed Jul 28 13:06:18 GMT +0000 2021
    Interface: ifl-0/0/1
    Outer VLAN: 1000
    Inner VLAN: 2002
    Client MAC: 02:00:00:00:00:04
    Username: blaster@l2tp.de
    Agent-Remote-Id: DEU.RTBRICK.2
    Agent-Circuit-Id: 0.0.0.0/0.0.0.0 eth 0:2
    Accounting-Session-Id: 72339069014638703:1627477569
    L2TP Disconnect Cause:
        Code: Normal disconnection (LCP terminate-request)
        Protocol: 0
        Direction: Peer
        Message: N/A
```

IPoE

The following commands are applicable for IPoE subscribers only.

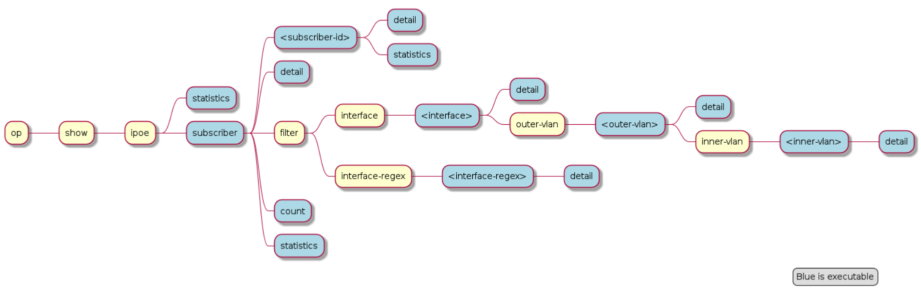


Figure 20. IPoE Operational Commands

```
supervisor@leaf1: op> show ipoe subscriber detail
Subscriber-Id           Interface        VLAN      MAC                  State
DHCPv4      DHCPv6
216454257090494465     ifl-0/0/1     8:1       02:00:00:00:00:01 ESTABLISHED
Bound       Bound
216454257090494466     ifl-0/0/1     8:2       02:00:00:00:00:02 ESTABLISHED
Bound       Bound
216454257090494467     ifl-0/0/1     8:3       02:00:00:00:00:03 ESTABLISHED
Bound       Bound
216454257090494468     ifl-0/0/1     8:4       02:00:00:00:00:04 ESTABLISHED
Bound       Bound
```

Further details per subscriber can be shown with the following command.

```
supervisor@leaf1: op> show ipoe subscriber 216454257090494465 detail
```

```
Subscriber-Id: 216454257090494465
    State: ESTABLISHED
    Uptime: Mon Jun 14 15:46:15 GMT +0000 2021 (0:02:19.421591)
    Interface: ifl-0/0/1
    Outer VLAN: 8
    Inner VLAN: 1
    Client MAC: 02:00:00:00:00:01
    Gateway Interface: lo-0/0/0/1
    Gateway Instance: default
    Gateway IPv4: 198.51.100.200/255.255.255.255
    Gateway MAC: 7a:52:4a:c0:00:01
    Agent-Remote-Id: DEU.RTBRICK.1
    Agent-Circuit-Id: 0.0.0.0/0.0.0.0 eth 0:1
    DHCPv4:
        Mode: Server
        State: Bound
        Address: 198.51.100.202/255.255.255.255
        Lease Created: Mon Jun 14 15:46:15 GMT +0000 2021 (0:02:19.427443)
        Lease Time: 300 seconds
        Lease Expire: 161 seconds
    DHCPv6:
        Mode: Server
        State: Bound
        Client DUID: 00030001020000000001
        Server DUID: 0003001b78524afffec00001
        IA_NA:
            Address: 2001:db8:0:96
            IAID: 1181407340
            Active: True
        IA_PD:
            Prefix: 2001:db8:0:333/32
            IAID: 4095128883
            Active: True
        Lease Created: Mon Jun 14 15:46:15 GMT +0000 2021 (0:02:19.428676)
        Lease Time (Lifetime): 14400 seconds
        Lease Expire: 14261 seconds
        Preferred Lifetime: 1800 seconds
```

Local Address Pools

Rather than using recommended IP addresses for technical documents, the document shows actual IP pool ranges.

The usage of local address pools can be monitored using the show subscriber pool commands as shown below.

```
supervisor@switch: op> show subscriber pool summary
Pool Name                          AFI  Usage          Range
pool-A                             IPv4 256/256        10.0.1.0 - 10.0.1.255
pool-B                             IPv4 2/256          10.0.2.0 - 10.0.2.255
pool-C                             IPv4 0/256          10.0.3.0 - 10.0.3.255
pool-D                             IPv4 0/256          10.0.4.0 - 10.0.4.255

supervisor@switch: op> show subscriber pool ipv4 pool-A
Pool Name: pool-A
    AFI: IPv4
    Usage: 256/256
    Range: 10.0.1.0 - 10.0.1.255
    Next: pool-B

supervisor@switch: op> show subscriber pool ipv4 pool-B
Pool Name: pool-B
    AFI: IPv4
    Usage: 2/256
    Range: 10.0.2.0 - 10.0.2.255
    Next: pool-C

supervisor@switch: op> show subscriber pool ipv4 pool-B allocation
Subscriber-Id          Timestamp                            Address/Prefix
72339069014638598      Wed Sep 15 09:02:15 GMT +0000 2021   10.0.2.0
72339069014638602      Wed Sep 15 09:02:15 GMT +0000 2021   10.0.2.1
```

5.1.4. Supported Standards

RFC and draft compliance are partial except as specified.

PPPoE

- RFC 1516
- RFC 1661 (partly)
- RFC 1332 (partly)
- RFC 5072 (partly)
- RFC 1334 (partly)
- RFC 4638

RADIUS

- RFC 2865 (partly)
- RFC 3162 (partly)
- RFC 2866 (partly)
- RFC 4372 (partly)
- RFC 2869 (partly)

DHCPv4

- RFC 951 (partly)
- RFC 1542 (partly)

- RFC 2131 (partly)
- RFC 2132 (partly)
- RFC 3046 (partly)

DHCPv6

- RFC 8415 (partly)

Access Line Information

The access line identification and characterization information are defined in the Broadband Forum (BBF) formerly known DSL Forum attributes including Agent-Remote-Id and Agent-Circuit-Id.

See the following references for more information about access line attributes.

- RFC 4679 DSL Forum Vendor-Specific RADIUS Attributes
- RFC 6320 ANCP (partly)
- Broadband Forum TR-101 (partly)
- draft-lihawi-ancp-protocol-access-extension-04 (partly)

L2TPv2

RFC and draft compliance are partial except as specified.

RFC 2661 - Layer Two Tunneling Protocol (L2TPv2)

RFC compliant L2TPv2 Access Concentrator (LAC) with the following protocol limitations:

- No support for LNS initiated outbound calls (OCRQ, OCRP and OCCN)
- No support for WAN-Error-Notify (WEN) Messages send by LAC to LNS
- No support for Set-Link-Info (SLI) Messages send by LNS to LAC
- No support for L2TP over IPv6
- No support for L2TP offset values other than 0.

RFC 5515 - L2TP Access Line Information AVP Extensions

- Support for access line AVP send (LAC) and received (LNS) as part of the L2TP Incoming-Call- Request (ICRQ) message.
- Response to Connect-Speed-Update-Request (CSURQ) L2TP messages is currently not supported.

RFC 2868 - RADIUS Attributes for Tunnel Protocol Support

RADIUS support for L2TP with the following limitations:

- No support of FQDN format for IP addresses
- No support Tunnel-Medium-Type other than IPv4

RFC 3145 - L2TP Disconnect Cause Information

Send meaningful disconnect cause information to LNS and display received disconnect cause information for tunnels and sessions.

Supported Hardware

You can find more detailed information about what RtBrick features are supported on each hardware platform, see the *Platform Guide*.

5.2. RBFS RADIUS Services

5.2.1. RADIUS Services Overview

The modular, scalable subscriber management that RtBrick calls the next generation access infrastructure (ng-access) provides support for protocols such as PPPoE, IPoE, L2TP and RADIUS.

The subscriber management infrastructure provides the next generation of internet access protocols designed for carrier grade services in regards to scalability and robustness. One of the challenges for carrier networks is interwork with numerous client devices which requires a well implemented, industry proven access protocol stack, including support for all relevant RFCs.

This implementation is designed to be a set of distributed services for increased scaling and stability. The subscriber management implementation has three components:

- subscriberd: subscriber management and AAA (local, RADIUS, and other support)
- pppoed: PPPoE/PPP session handling
- l2tpd: L2TP tunnel and session handling

The subscriber daemon (subscriberd) is the central application, keeping the current subscriber state as well as being responsible for Authentication, Authorization and Accounting (AAA).

This document describes the subscriber management RADIUS service implementation on RBFS. The term subscriber describes an access user or session from a higher level decoupled from underlying protocols like PPPoE or IPoE. Subscribers in RBFS can be managed locally or via RADIUS where this document considers RADIUS only. Each subscriber is uniquely identified by a 64bit number called subscriber-id. Remote Authentication Dial-In User Service (RADIUS) is a networking protocol that provides centralized Authentication, Authorization and Accounting (AAA) management for all types of subscribers (PPPoE, or IPoE). RADIUS servers can perform as authentication and accounting servers or change of authorization (CoA) clients. Authentication servers maintain authentication records for subscribers. The subscriber daemon requests authentication in RADIUS access-request messages before permitting subscribers access. Accounting servers handle accounting records for subscribers. The subscriber daemon transmits RADIUS accounting-start, interim and stop messages to the servers. Accounting is the process of tracking subscriber activity and network resource usage in a subscriber session. This includes the session time called time accounting and the number of packets and bytes transmitted during the session called volume accounting.

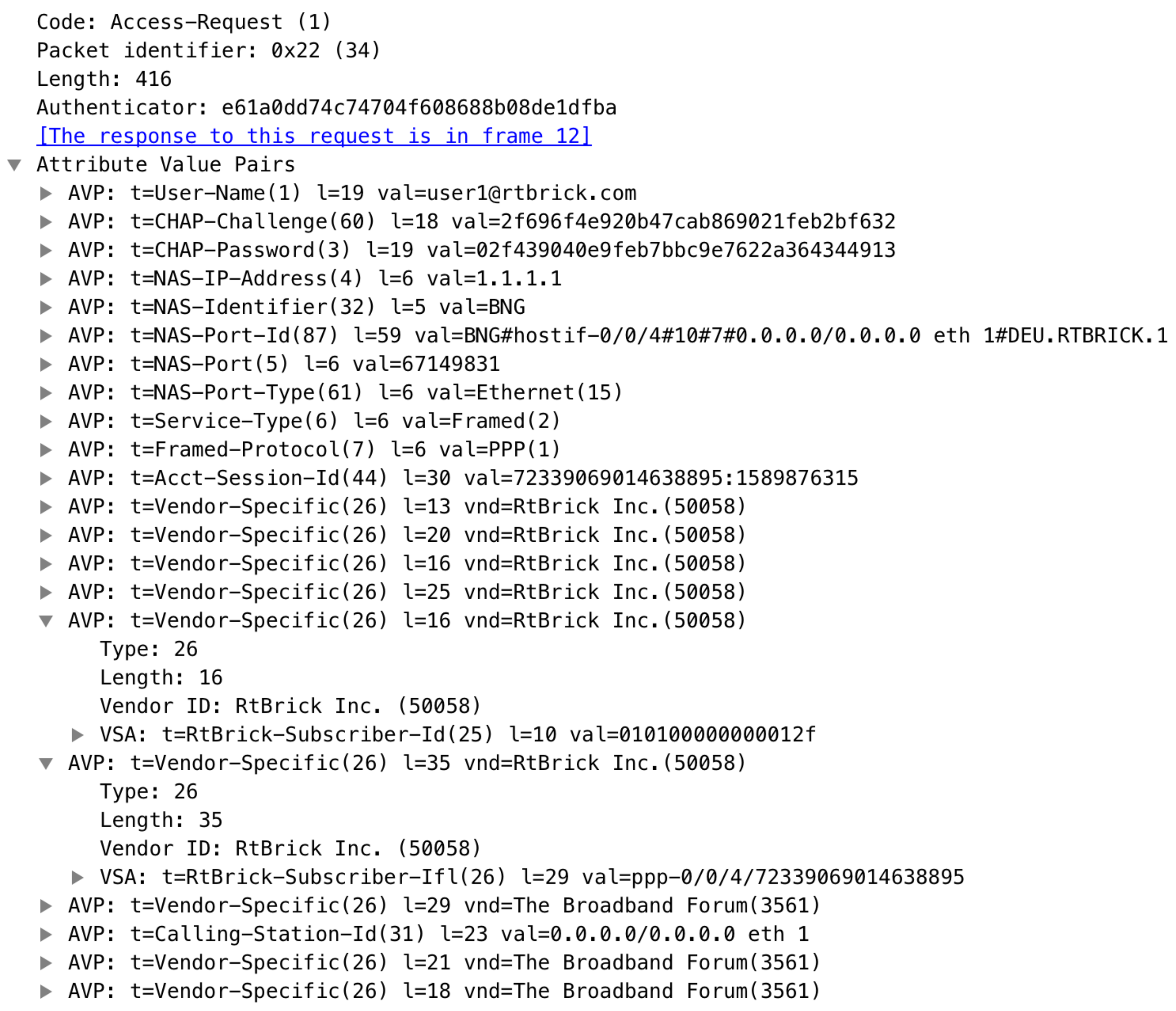
A RADIUS server can behave as a change of authorization (CoA) client allowing dynamic changes for subscriber sessions. The subscriber daemon supports both RADIUS CoA messages and disconnect messages. CoA messages can modify the characteristics of existing subscriber sessions without loss of service, disconnect messages can terminate subscriber sessions.

Each RADIUS request from subscriber daemon includes the RADIUS accounting-session-id attribute (type 44) with a format which is configurable in the AAA profile and includes at least the subscriber-id to identify the corresponding subscriber.

The default format (<subscriber-id>.<timestamp>) includes also an Unix timestamp to ensure that the tuple of NAS-Identifier (e.g. hostname) and Accounting-Session-Id is global unique to be usable as key in RADIUS databases.

Additionally to subscriber-id and accounting-session-id each subscriber consists also of a subscriber-ifl build based on physical port information and subscriber-id (ifp: ifp-0/0/1 and subscriber-id: 72339069014638610 subscriber-ifl: ppp-0/0/1/72339069014638610) which is required as handle in the RBFS forwarding infrastructure. All those three informations are part of the RADIUS access-request message:

- accounting-session-id: standard attribute 44
- subscriber-id: RtBrick VSA (26-50058-25 RtBrick-Subscriber-Id)
- subscriber-ifl: RtBrick VSA (26-50058-26 RtBrick-Subscriber-Ifl)



```
  Code: Access-Request (1)
  Packet identifier: 0x22 (34)
  Length: 416
  Authenticator: e61a0dd74c74704f608688b08de1dfba
  [The response to this request is in frame 12]
▼ Attribute Value Pairs
  ▶ AVP: t=User-Name(1) l=19 val=user1@rtbrick.com
  ▶ AVP: t=CHAP-Challenge(60) l=18 val=2f696f4e920b47cab869021feb2bf632
  ▶ AVP: t=CHAP-Password(3) l=19 val=02f439040e9feb7bbc9e7622a364344913
  ▶ AVP: t=NAS-IP-Address(4) l=6 val=1.1.1.1
  ▶ AVP: t=NAS-Identifier(32) l=5 val=BNG
  ▶ AVP: t=NAS-Port-Id(87) l=59 val=BNG#hostif-0/0/4#10#7#0.0.0.0/0.0.0.0 eth 1#DEU.RTBRICK.1
  ▶ AVP: t=NAS-Port(5) l=6 val=67149831
  ▶ AVP: t=NAS-Port-Type(61) l=6 val=Ethernet(15)
  ▶ AVP: t=Service-Type(6) l=6 val=Framed(2)
  ▶ AVP: t=Framed-Protocol(7) l=6 val=PPP(1)
  ▶ AVP: t=Acct-Session-Id(44) l=30 val=72339069014638895:1589876315
  ▶ AVP: t=Vendor-Specific(26) l=13 vnd=RtBrick Inc.(50058)
  ▶ AVP: t=Vendor-Specific(26) l=20 vnd=RtBrick Inc.(50058)
  ▶ AVP: t=Vendor-Specific(26) l=16 vnd=RtBrick Inc.(50058)
  ▶ AVP: t=Vendor-Specific(26) l=25 vnd=RtBrick Inc.(50058)
  ▼ AVP: t=Vendor-Specific(26) l=16 vnd=RtBrick Inc.(50058)
      Type: 26
      Length: 16
      Vendor ID: RtBrick Inc. (50058)
    ▶ VSA: t=RtBrick-Subscriber-Id(25) l=10 val=010100000000012f
  ▼ AVP: t=Vendor-Specific(26) l=35 vnd=RtBrick Inc.(50058)
      Type: 26
      Length: 35
      Vendor ID: RtBrick Inc. (50058)
    ▶ VSA: t=RtBrick-Subscriber-Ifl(26) l=29 val=ppp-0/0/4/72339069014638895
  ▶ AVP: t=Vendor-Specific(26) l=29 vnd=The Broadband Forum(3561)
  ▶ AVP: t=Calling-Station-Id(31) l=23 val=0.0.0.0/0.0.0.0 eth 1
  ▶ AVP: t=Vendor-Specific(26) l=21 vnd=The Broadband Forum(3561)
  ▶ AVP: t=Vendor-Specific(26) l=18 vnd=The Broadband Forum(3561)
```

The subscriber-id is an unsigned 64bit integer which is shown as a hex number in Wireshark.

Each subscriber is formed based on configuration profiles and individual settings retrieved via RADIUS. Conflicts between RADIUS defined attributes and profile attributes are solved by prioritizing those received from RADIUS which is common best practices for broadband access concentrators. New subscribers are signalled via RADIUS access-request and either accepted by RADIUS access-accept or rejected by RADIUS access-reject message from RADIUS server. The RADIUS access-accept includes all attributes required to form the subscriber like IP addresses, DNS servers and referenced configuration profiles. Some of those attributes can be changed by RADIUS dynamically using CoA requests without disconnecting the subscriber.

Some of those attributes refer to RADIUS services which are described in detail in this document. The term RADIUS services described the ability to dynamically control the properties of a session via RADIUS used by access providers to build their different products and services based on their network infrastructure. This is used to control QoS settings like shaper rates dynamically based on changed line quality or possible volume quotas. It is also used to dynamically enable or disable services like Voice or IPTV on a per subscriber basis.

The RADIUS accounting accuracy should comply with §45g of the German Telecommunications Act (TKG) further named by TKG§45. This applies for time and volume based products and services. Products or services with unlimited volumes but with changed properties after a certain amount of traffic are considered as volume based products as well. One example here is a fair use policy which enforces a rate limit after a certain volume. The requirements for time based accounting apply only to products or services which are charged by time or changed properties after a certain time which is not very common in the market today.

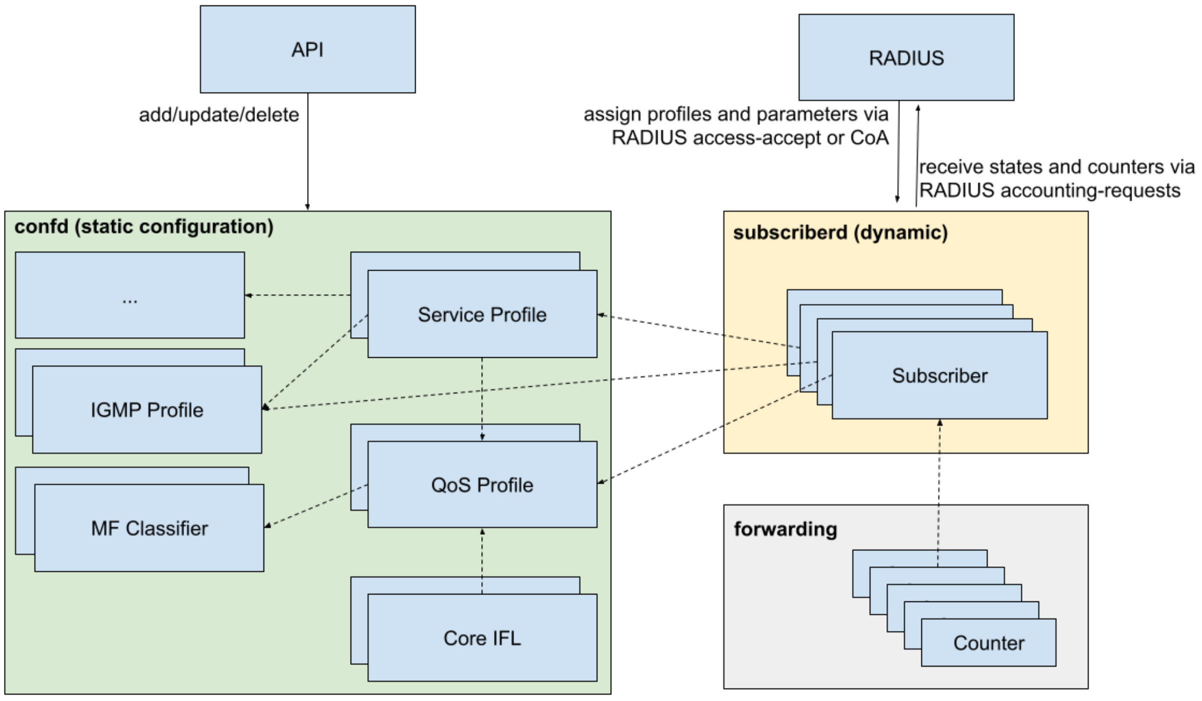
Supported Platforms

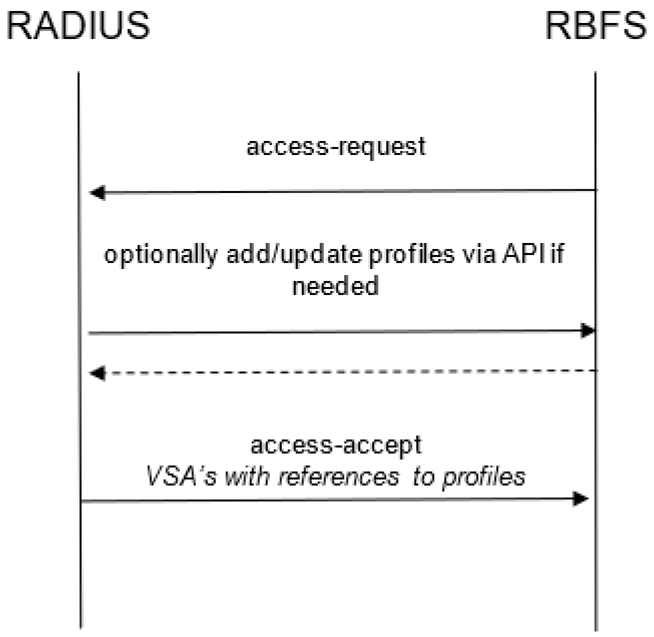
Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

5.2.2. RADIUS Service Solution Overview

The general concept of RADIUS services in RtBrick FullStack (RBFS) consists of pre-configured profiles in the global static configuration assigned to dynamic subscribers via RADIUS VSA in access-accept and CoA request messages. The profiles are managed as any other configuration in RBFS via corresponding API which is not explained here in detail.

The following pictures show the concept of global static configurations referenced by RADIUS controlled dynamic subscribers.


API
add/update/delete
RADIUS
assign profiles and parameters via RADIUS access-accept or CoA
receive states and counters via RADIUS accounting-requests
confd (static configuration)
...
Service Profile
IGMP Profile
QoS Profile
MF Classifier
Core IFL
subscriberd (dynamic)
Subscriber
forwarding
Counter


RADIUS
RBFS
access-request
optionally add/update profiles via API if needed
access-accept
VSA's with references to profiles

The required profiles and filters should be present before service becomes active which can be done immediately after start-up or between access-request and access-accept as shown in the flow diagram on the left.

5.2.3. RADIUS Control

This chapter explains the subscriber and service-related RBFS extensions and RtBrick VSA related to RADIUS services.

RADIUS Session and Idle Timeout

The session and idle timeout values are initialized with zero, which means infinity or disabled. Those values will be optionally overwritten with the values in AAA configuration profile and RADIUS access-accept if present. The priority is RADIUS attributes over AAA configuration profile.

The session (attribute 27) and idle (attribute 28) timeout RADIUS attributes are defined in RFC 2865.

RFC and draft compliance are partial except as specified.

The idle timeout is based on outgoing logical interface statistics (OutLIF) for the subscriber-ifl to determine subscriber inactivity.

RADIUS Routing Instance

VSA 26-50058-137 - RtBrick-Instance

This attribute is applicable to non-IPoE subscribers only. It allows to change the routing instance through a RADIUS access-accept message.

For IPoE subscribers, the routing instance is automatically determined based on the gateway IFL.

RADIUS Gateay-Ifl

VSA 26-50058-138 - RtBrick-Gateway-Ifl

This attribute is applicable to IPoE subscribers only. It allows to change the actual gateway IFL. Consequently, this also indirectly modifies the routing instance through a RADIUS Access-Accept message.

RADIUS DNS Server

VSA 26-50058-131 - RtBrick-DNS-Primary

VSA 26-50058-132 - RtBrick-DNS-Secondary

VSA 26-50058-134 - RtBrick-DNS-Primary-IPv6

VSA 26-50058-135 - RtBrick-DNS-Secondary-IPv6

Those attributes allow to set the primary and secondary IPv4 and IPv6 DNS servers via RADIUS access-accept. DNS servers already set via access configuration profile will be overwritten by RADIUS.

RADIUS Service Profile

VSA 26-50058-70 - RtBrick-Service-Profile

Subscribers can be associated with a service-profile which defines the actual service properties like QoS or IGMP profiles and configuration settings. The service profile can be assigned or changed using the RtBrick-Service-Profile VSA.

```
RtBrick-Service-Profile = <service-profile>
```

This attribute is supported via RADIUS access-accept and CoA requests.

All dynamic QoS settings like shaper and policer rates will be reset if the new service-profile includes a qos-profile attribute also if active qos-profile and old qos-profile is equal. All QoS settings remain unchanged if the referenced service profile does not include the qos-profile attribute. If the referenced service profile updates the qos-profile attribute and additional shaper or policer rates are included in the same CoA request which updates the service-profile, those shaper and policer settings will be applied to the new QoS configuration profile after reset. This means that we reset all incremental changes done before. In example if Voice shaper rate has changed to another value, after profile change the default value from profile is used.

RADIUS CoA requests referencing a service profile which is not found on device will be rejected with CoA-NAK (invalid-request) but without changing any subscriber QoS settings. This behavior is different for RADIUS access-accept where service profiles are just ignored if not found. In both cases a warning is generated in subscriber daemon log if a referenced service profile is not found.

RADIUS AAA Profile

VSA 26-50058-69 - RtBrick-AAA-Profile

This attribute allows to change the associated AAA profile from RADIUS access-accept. This is primarily used to change the accounting adjustment values which are defined in this profile. Simple example here is to use different adjustment values for L2TP and local terminated PPPoE sessions. Another valid use case is to assign different RADIUS accounting servers for in example L2TP sessions or wholesale customers.

```
RtBrick-AAA-Profile = <aaa-profile>
```

This attribute is supported via RADIUS access-accept only.

RADIUS QoS Profile

VSA 26-50058-62 - RtBrick-QoS-Profile

```
RtBrick-QoS-Profile = <qos-profile>
```

This attribute is supported by RADIUS access-accept and CoA request. The QoS configuration profile can be either selected via service-profile or directly using this attribute which has priority of the service-profile.

All dynamic QoS settings like MFC, queue sizes, shaper and policer rates will be reset if this attribute is present in a CoA request also if new qos-profile and old qos-profile is equal. If additional shaper or policer rates are included in the same CoA request which updates the service-profile, those shaper and policer settings will be applied to the new QoS configuration after reset.

The subscriber management infrastructure does not check if a referenced QoS profile exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the profile was added. In the meantime there is no QoS configuration applied.

RADIUS QoS Parent Scheduler

VSA 26-50058-64 - RtBrick-QoS-Parent-Scheduler

The parent scheduler element of the scheduler-map assigned to the subscriber can be selected with this attribute. If not present, the scheduler-map will be directly bound to the local IFP where the session is established.

This attribute is supported in RADIUS access-accept only (no CoA support) and will set the parent scheduler element of the subscriber.

> Providing a QoS parent scheduler which is not present on the corresponding IFP will lead to black howling of all egress data traffic. Control traffic is not impacted and therefore the session will remain.

RADIUS QoS Shaper

VSA 26-50058-63 - RtBrick-QoS-Shaper

Subscribers can be associated with a QoS profile which is assigned via service-profile or directly via corresponding VSA (RtBrick-QoS-Profile). This profile is used to instantiate the QoS resources for the subscriber including schedulers, queues and shapers. The assigned shaper instances can be updated using the RtBrick-QoS-Shaper VSA (attribute 26-50058-63) which will apply to the QoS instance of the corresponding subscriber only, but not to the other subscribers using the same QoS profile. It is only possible to update existing shapers dynamically but it is not possible to create a new shaper via RADIUS.

The RtBrick-QoS-Shaper value is a string which contains a list of multiple shaper settings separated by semicolon. Each shaper setting contains a shaper name, high flow rate and low flow rate separated by comma. The actual shaper rate is the sum of high and low flow rate.

```
RtBrick-QoS-Shaper = <shaper-name>,<high-kbps>,<low-kbps>;<shaper-name>,...
```

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
RtBrick-QoS-Shaper = name=<shaper-name>,high=<high-kbps>,low=<low-kbps>;...
```

This attribute is supported by RADIUS access-accept and CoA request.

Updating a single shaper (e.g. voice_shaper) via CoA does not require to include other shapers meaning that only the shapers included in the attribute will be updated and all other shapers remain unchanged.

Assume the following example which adapts the session and voice shaper instance of a subscriber.

```
supervisor@rtbrick: op> show config forwarding-options class-of-service shaper
shaper_session
{
  "rtbrick-config:shaper": {
    "shaper-name": "shaper_session",
    "shaping-rate-high": 48000,
    "shaping-rate-low": 2000
  }
}
supervisor@rtbrick: op> show config forwarding-options class-of-service shaper
shaper_voice
{
  "rtbrick-config:shaper": {
    "shaper-name": "shaper_voice",
    "shaping-rate-high": 1000,
    "shaping-rate-low": 0
  }
}
```

RADIUS VSA Change Session Shaper Only

```
RtBrick-QoS-Shaper: shaper_session,14000,2000
```

RADIUS VSA Change Session and Voice Shaper

```
RtBrick-QoS-Shaper: shaper_session,14000,2000;shaper_voice,2000
```

All active dynamic shapers are stored in the table **global.access.1.subscriber.qos.shaper** to handover those information to forwarding infrastructure. This table can be used to verify the dynamic shapers which are active for a given subscriber.

The CLI command show subscriber <id> qos displays those information in a nice human readable format.

```
supervisor@rtbrick: op> show subscriber 72339069014638610 qos
Subscriber-Id: 72339069014638610
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/1/72339069014638610
    Profile: qos_profile
    Parent: pon1
    Dynamic Shaper: shaper_voice
        Rate Low: 0 kbps
        Rate High: 2000 kbps
    Dynamic Shaper: shaper_session
        Rate Low: 2000 kbps
        Rate High: 14000 kbps
```

A shaper rate of 0 means infinity!

RADIUS QoS Policer

VSA 26-50058-65 - RtBrick-QoS-Policer

The RtBrick-QoS-Policer value is a string which contains a list of multiple policer level settings separated by semicolon. Each setting contains a level, cir, cbs, pir, pbs, max-cir and max-pir separated by comma.

RtBrick-QoS-Policer = <level>,<cir>,<cbs>,<pir>,<pbs>,<max-cir>,<max-pir>;<level>…
 	
Example:
RtBrick-QoS-Policer = 1,2000,200;2,8000,800;3,0,800;4,0,800

- **level**: 1 (highest priority) to 4 (lowest priority)
- **cir**: Ingress policer committed information rate (kbps)
- **cbs**: Ingress policer committed burst size (kbits)
- **pir**: Ingress policer peak information rate (kbps)
- **pbs**: Ingress policer peak burst size (kbits)
- **max-cir**: max ingress policer committed information rate (kbps)
- **max-pir**: max ingress policer peak information rate (kbps)

If PIR and PBS are not defined, the values from CIR and CBS are used as PIR and PBS as well. The max CIR and max PIR attributes are optional default set to unlimited.

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

RtBrick-QoS-Policer = level=<level>,cir=<cir>,cbs=<cbs>,pir=<pir>,pbs=<pbs>,max-cir=<max-cir>,max-pir=<max-pir>;...

Example:
RtBrick-QoS-Policer = level=4,cir=1m,cbs=256;cir=2m,cbs=512,level=3

This attribute is supported by RADIUS access-accept and CoA request.

Updating a single policer level via CoA does not require to include other levels meaning that only the levels included in the attribute will be updated and all other levels remain unchanged. It is only possible to update existing policer levels dynamically but it is not possible to create a new level via RADIUS.

Assume the following example which adapts the just one level as well as all levels of a subscriber.

```
supervisor@rtbrick: op> show config forwarding-options class-of-service policer
policer-residential
{
  "rtbrick-config:policer": {
    "policer-name": "policer-residential",
    "level1-rates": {
      "cir": 1000,
      "cbs": 100,
      "pir": 1000,
      "pbs": 100
    },
    "level2-rates": {
      "cir": 4000,
      "cbs": 400,
      "pir": 4000,
      "pbs": 400
    },
    "level3-rates": {
      "cir": 0,
      "cbs": 800,
      "pir": 0,
      "pbs": 800
    },
    "level4-rates": {
      "cir": 0,
      "cbs": 800,
      "pir": 0,
      "pbs": 800
    },
    "levels": 4,
    "type": "two-rate-three-color"
  }
}
```

RADIUS VSA Change Level 2 Only

```
RtBrick-QoS-Policer: 2,12000,1200
```

RADIUS VSA Change all Levels

```
RtBrick-QoS-Policer: 1,2000,200;2,8000,800;3,0,800;4,0,800
```

All active dynamic policer level settings are stored in the table **global.access.1.subscriber.qos** to handover those information to forwarding infrastructure. This table can be also used to verify the dynamic policer settings for a given subscriber.

The CLI command show subscriber <id> qos displays those information in a nice human readable format.

```
supervisor@rtbrick: op> show subscriber 72339069014638610 qos
Subscriber-Id: 72339069014638610
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/1/72339069014638610
    Profile: qos_profile
    Parent: pon1
    Dynamic Ingress Policer Level 1:
        CIR: 2000 kbps CBS: 200 kbps
        PIR: 2000 kbps PBS: 200 kbps
    Dynamic Ingress Policer Level 2:
        CIR: 8000 kbps CBS: 800 kbps
        PIR: 8000 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 3:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 4:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Shaper: shaper_voice
        Rate Low: 0 kbps
        Rate High: 2000 kbps
    Dynamic Shaper: shaper_session
        Rate Low: 2000 kbps
        Rate High: 14000 kbps
```

RADIUS QoS MFC

VSA 26-50058-66 - RtBrick-QoS-MFC

The multifield classifier can be either derived from qos-profile or directly using this attribute which has priority of the qos-profile.

```
RtBrick-QoS-MFC = <mfc-name>
```

This attribute is supported by RADIUS access-accept and CoA request.

The subscriber management infrastructure does not check if a referenced multifield classifier exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the classifier was added.

The string received in these attributes should be stored as mf_classifier_name in the RADIUS attribute object and as well as in the corresponding subscriber-qos object.

RADIUS QoS Queues

VSA 26-50058-67 - RtBrick-QoS-Queues

Subscribers can be associated with a QoS profile which is assigned via service-profile or directly via corresponding VSA (RtBrick-QoS-Profile). This profile is used to instantiate the QoS resources for the subscriber including schedulers, queues and shapers. The assigned queue instances can be updated using the RtBrick-QoS-Queues VSA (attribute 26-50058-67) which will apply to the QoS instance of the corresponding subscriber only, but not to the other subscribers using the same QoS profile. It is only possible to update existing queues dynamically but it is not possible to add queues via RADIUS.

The RtBrick-QoS-Queues value is a string which contains a list of multiple queue settings separated by semicolon. Each queue setting contains a queue name and queue size in bytes separated by comma.

```
RtBrick-QoS-Queues = <queue-name>,<size-bytes>;<queue-name>,<size-bytes>;...
```

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
RtBrick-QoS-Queues = name=<queue-name>,size=<size-bytes>;name=...
```

This attribute is supported by RADIUS access-accept and CoA request.

Updating a single queue (e.g. best-effort) via CoA does not require to include other queues meaning that only the queues included in the attribute will be updated and all other queues remain unchanged.

The subscriber management infrastructure does not check if a referenced queue exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the queue was added.

All active dynamic queues are stored in the table **global.access.1.subscriber.qos.queue** to handover those information to forwarding infrastructure. This table can be also used to verify the dynamic queues which are active for a given subscriber.

The CLI command show subscriber <id> qos displays those information in a nice human readable format.

```
supervisor@rtbrick: op> show subscriber 72339069014638610 qos
Subscriber-Id: 72339069014638610
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/1/72339069014638610
    Profile: qos_profile
    Parent: pon1
    Dynamic Ingress Policer Level 1:
        CIR: 2000 kbps CBS: 200 kbps
        PIR: 2000 kbps PBS: 200 kbps
    Dynamic Ingress Policer Level 2:
        CIR: 8000 kbps CBS: 800 kbps
        PIR: 8000 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 3:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Ingress Policer Level 4:
        CIR: 0 kbps CBS: 800 kbps
        PIR: 0 kbps PBS: 800 kbps
    Dynamic Shaper: shaper_voice
        Rate Low: 0 kbps
        Rate High: 2000 kbps
    Dynamic Shaper: shaper_session
        Rate Low: 2000 kbps
        Rate High: 14000 kbps
    Dynamic Queue: voice
        Size: 200000 byte
```

RADIUS IGMP Attributes

IGMP service can be dynamically enabled on a subscriber using Radius IGMP Attributes. These attributes are supported by RADIUS access-accept and CoA requests.

Following IGMP related attributes are supported:

VSA 26-50058-71 - RtBrick-IGMP-Status

This attribute can dynamically enable/disable IGMP for a subscriber

Value	Code	Description
DISABLED	0	Disable IGMP
ENABLED	1	Enable IGMP

VSA 26-50058-72 - RtBrick-IGMP-Profile

```
RtBrick-IGMP-Profile = <igmp-profile>
```

This attribute specifies the IGMP-profile to be associated with the subscriber. IGMP profile is configured locally in IGMP with all the IGMP related attributes. This attribute can dynamically associate such a locally configured profile to a subscriber

The subscriber management infrastructure does not check if a referenced IGMP profile exists or not. This is handled by IGMP infrastructure which continues processing the subscriber IGMP settings as soon as the profile was added.

VSA 26-50058-73 - RtBrick-IGMP-Version

This attribute can specify the version of IGMP for a subscriber.

Value	Code	Description
V1	1	IGMP Version 1 (not supported)
V2	2	IGMP Version 2
V3	3	IGMP Version 3 (Default version if this attribute is not set)

VSA 26-50058-74 - RtBrick-IGMP-Max-Members

This integer attribute can specify the number of parallel streaming (maximum IGMP membership) for a subscriber.

The CLI command show subscriber <id> igmp displays those information.

```
supervisor@rtbrick: op> show subscriber 72339069014638706 igmp
Subscriber-Id: 72339069014638706
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
    IFL: ppp-0/0/3/72339069014638706
    State: active
    Version: IGMPv3
    Profile: iptv-basic
    Max Members: 6
```

The states are active, inactive and disabled. The state is inactive if IGMP is enabled but the IPv4 protocol state is not active.

RADIUS Connection Status Message Attribute

VSA 26-50058-139 - RtBrick-Connection-Status-Message

This attribute allows to send a connection status message string via PPP LCP vendor extension (RFC2513) to the client.

The connection status message is typically used to inform clients about the available bandwidth via RADIUS access-accept or CoA request.

```
RtBrick-Connection-Status-Message = <string>
```

Adding or changing the connection status message triggers to send a LCP vendor specific packet with OUI set to f4:1e:5e (RtBrick Inc.) and kind 0x01.

RFC 2513 - Vendor Specific Packet

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Magic-Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      OUI                      |     Kind      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Value(s) ...
+-+-+-+-+-+-+-+-+
```

The value is encoded as TLV with type set to 0x01 and length including type and length field.

The actual status message string is limited to 247 bytes.

Connection-Status-Message Option

```
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |     Type      |    Length     | Connection-Status-Message     |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | ...
 +-+-+
```

PPP clients with support for connection status message will respond with a vendor specific packet of kind 0x02 and the same OUI but no value to acknowledge the packet. All other clients either ignore or reject the message using PPP LCP code reject.

The connection status message will be retried 10 times with an interval of 3 seconds until the client has acknowledged or rejected the packet.

The status of the connection status message can be verified with the following command.

```
supervisor@leaf1: op> show pppoe session 72339069014638706 detail
Subscriber-Id: 72339069014638706
    State: ESTABLISHED
    Uptime: Mon Sep 13 09:29:42 GMT +0000 2021 (0:14:57.625027)
    Interface: ifp-0/0/1
    Outer VLAN: 128
    Inner VLAN: 7
...
    PPP LCP:
        State: OPENED
        Negotiated Protocols: PAP, IPCP, IP6CP
        Negotiated Parameters: MRU, AUTH, MAGIC
        Magic Number: 130827225 Peer: 2835676915
        MRU: 1492 Peer: 1492
        Echo Interval: 30 seconds
    PPP LCP Connection Status Message:
        State: ACCEPTED
        Message: SRU=10000#SRD=100000#
...
```

RADIUS Ascend-Data-Filter Attribute

VSA 26-529-242 - Ascend-Data-Filter

The Ascend-Data-Filter attribute describes per subscriber filter terms in a simple binary format as described in the following table. Multiple of those attributes in a single RADIUS access-accept or CoA request message are combined to a dynamic filter where each attribute itself is one filter term. The order of those attributes in the RADIUS message is used to order the corresponding terms in the filter. This means that the first filter in the RADIUS packet has priority over the next and or last filter in the RADIUS packet.

Changes in such a filter via CoA requires that all attributes of the new filter must be sent. Therefore adding a single filter term requires sending the existing filter terms plus the new one or general speaking sending the intended target filter.

This filter is installed as a global packet filter placed before policer in ingress and before scheduler in egress to prevent that traffic dropped here is counted in accounting or consuming rate credits.

Field	Bytes	Values	Comments
Type	1	0 = ignore 1 = IPv4 3 = IPv6	
Action	1	0 = discard 1 = accept 0x20 = redirect	
Direction	1	0 = egress 1 = ingress	
Reserved	1	0	

Source
 	
Address
	IPv4 = 4
	
	IPv6 = 16
		source address	Match on source address is not supported for filters in ingress direction and automatically replaced with the subscriber addresses.

Field	Bytes	Values	Comments

Destination
 	
Address
	IPv4 = 4
	
	IPv6 = 16
		destination address	Match on destination address is not supported for filters in egress direction and automatically replaced with the subscriber addresses.

Source Prefix Length	1	0 = ignore prefix length	The number of leading one bits in the source address mask. Specifies the bits of interest.
Destination Prefix Length	1	0 = ignore prefix length	The number of leading one bits in the destination address mask. Specifies the bits of interest.
Protocol	1	0 = ignore IPv4 protocol number IPv6 next header	
Reserved	1	0	
Source Port	2	port number	UDP/TCP source port in network byte order (big endian)
Destination Port	2	port number	UDP/TCP destination port in network byte order (big endian)

Field	Bytes	Values	Comments
Source Port Qualifier	1	0 = no compare 1 = less than 2 = equal to 3 = greater than 4 = not equal to	The options 1, 3 and 4 are not implemented and mapped to 0 if received.
Destination Port Qualifier	1	0 = no compare 1 = less than 2 = equal to 3 = greater than 4 = not equal to	The options 1, 3 and 4 are not implemented and mapped to 0 if received.
Not Used	0 - N	0	Trailing bytes after destination port qualifier ignored.

The filter can be deleted dynamically by sending a single Ascend-Data-Filter attribute with a zero value.

Example:

```
Ascend-Data-Filter =
0x01000100000000000a0afffe00200000000000000000000000000000000000

    01          IPv4            // type
    00          discard         // action
    01          ingress         // direction
    00          -               // reserved
    00000000    -               // source address
    0a0afffe    198.51.100.101  // destination address
    00          -               // source prefix length
    20          /32             // destination prefix length
    00          -               // protocol
    00          -               // reserved
    0000        -               // source port
    0000        -               // destination port
    00          -               // source port qualifier
    00          -               // destination port qualifier
    00000...    -               // ignored
```

For ingress filters, it is not permitted to match based on the source address which is automatically replaced with the subscriber addresses. A similar limitation is valid for egress filters matching on destination address which is also replaced with subscriber addresses.

A new proprietary filter action 0x20 (32) is added for the redirect service. This can be used to enable HTTP Redirect for a subscriber using the ADF.

All active subscriber filters are stored in the table **global.access.1.subscriber.filter** to hand over that information to the forwarding infrastructure. This table can be also used to verify the filters which are active for a given subscriber.

The CLI command show subscriber <id> acl displays those filters in a nice human-readable format.

RADIUS Access Line Attributes

The access line identification and characterization information are defined in the Broadband Forum (BBF) formerly known DSL Forum attributes including Agent-Remote-Id and Agent-Circuit-Id.

See the following references for more information about access line attributes.

- RFC 4679 DSL Forum Vendor-Specific RADIUS Attributes
- RFC 6320 ANCP
- Broadband Forum TR-101
- draft-lihawi-ancp-protocol-access-extension-04

Those attributes will be sent in RADIUS access and accounting requests to the RADIUS server if learned from the underlying access protocol like PPPoE (BBF TR-101).

RBFS also provides the possibility to set those values via RADIUS by using the same attributes in the RADIUS access-accept or CoA request message.

Access Line Attributes Supported (RADIUS, PPPoE and L2TP):

Attribute	Name
26-3561-1	Agent-Circuit-Id
26-3561-2	Agent-Remote-Id
26-3561-18	PON-Access-Line-Attributes
26-3561-129	Actual-Data-Rate-Upstream
26-3561-130	Actual-Data-Rate-Downstream
26-3561-131	Minimum-Data-Rate-Upstream
26-3561-132	Minimum-Data-Rate-Downstream
26-3561-133	Attainable-Data-Rate-Upstream
26-3561-134	Attainable-Data-Rate-Downstream
26-3561-135	Maximum-Data-Rate-Upstream
26-3561-136	Maximum-Data-Rate-Downstream
26-3561-137	Minimum-Data-Rate-Upstream-Low-Power
26-3561-138	Minimum-Data-Rate-Downstream-Low-Power
26-3561-139	Maximum-Interleaving-Delay-Upstream
26-3561-140	Actual-Interleaving-Delay-Upstream
26-3561-141	Maximum-Interleaving-Delay-Downstream
26-3561-142	Actual-Interleaving-Delay-Downstream
26-3561-144	Access-Loop-Encapsulation
26-3561-145	DSL-Type
26-3561-151	PON-Access-Type
26-3561-155	Expected-Throughput-Upstream
26-3561-156	Expected-Throughput-Downstream
26-3561-157	Attainable-Expected-Throughput-Upstream
26-3561-158	Attainable-Expected-Throughput-Downstream

Attribute	Name
26-3561-159	Gamma-Data-Rate-Upstream
26-3561-160	Gamma-Data-Rate-Downstream
26-3561-161	Attainable-Gamma-Data-Rate-Upstream
26-3561-162	Attainable-Gamma-Data-Rate-Downstream
26-3561-176	ONT-ONU-Average-Data-Rate-Downstream
26-3561-177	ONT-ONU-Peak-Data-Rate-Downstream
26-3561-178	ONT-ONU-Maximum-Data-Rate-Upstream
26-3561-179	ONT-ONU-Assured-Data-Rate-Upstream
26-3561-180	PON-Tree-Maximum-Data-Rate-Upstream
26-3561-181	PON-Tree-Maximum-Data-Rate-Downstream

This table includes already the new attributes defined in draft-lihawi-ancp-protocol-access-extension-04 which results in possible changes in case the draft is updated.

Changes in at least one of those line attributes via CoA trigger a RADIUS interim accounting request with the new values.

Changes in actual data rate upstream/downstream via CoA request may trigger the L2TP LAC to send a CSUN request with updated connection speed to the LNS depending on the actual L2TP configuration.

RADIUS L2TP

Tunneling of PPPoE sessions via L2TPv2 (L2TP LAC) is supported on RBFS and can be controlled via RADIUS as described in RFC 2868 RADIUS Attributes for Tunnel Protocol Support with some limitations:

- No support of FQDN format for IP addresses
- No support Tunnel-Medium-Type other than IPv4

To establish a PPPoE session via L2TP, the tunnel-type must be configured as L2TP. This configuration can be achieved either for local users or by utilizing the corresponding tunnel-type attribute through RADIUS.

Defining an L2TP configuration profile is essential, which can be referenced through an access configuration profile or by employing the appropriate RADIUS VSA. The actual tunnels may either be defined locally via an L2TP pool configuration or set up dynamically through RADIUS.

```
# FreeRADIUS
# RADIUS authenticated with local defined tunnels.
"radius@l2tp" Cleartext-Password := "test"
    Service-Type = Framed-User,
    Framed-Protocol = PPP,
    Tunnel-Type:0 = L2TP

# RADIUS authenticated with dynamic tunnels.
"tunnel@rl2tp" Cleartext-Password := "test"
    Service-Type = Framed-User,
    Framed-Protocol = PPP,
    Tunnel-Type:1 = L2TP,
    Tunnel-Client-Endpoint:1 = "192.0.2.1",
    Tunnel-Server-Endpoint:1 = "192.0.2.2",
    Tunnel-Client-Auth-Id:1 = "bng",
    Tunnel-Server-Auth-Id:1 = "lns",
    Tunnel-Preference:1 = 100,
    Tunnel-Password:1 = "test",
    Tunnel-Type:2 += L2TP,
    Tunnel-Client-Endpoint:2 += "192.0.2.3",
    Tunnel-Server-Endpoint:2 += "192.0.2.4",
    Tunnel-Client-Auth-Id:2 += "BNG",
    Tunnel-Server-Auth-Id:2 += "LNS2",
    Tunnel-Preference:2 += 200,
    Tunnel-Password:2 += "test"
```

In addition to the standard attributes, the following vendor-specific attributes are supported for L2TP.

RFC and draft compliance are partial except as specified.

VSA 26-50058-40 - RtBrick-L2TP-Pool

Instead of RADIUS tunnel attributes it is also possible to configure local L2TP tunnel pools and assign them with this attribute.

VSA 26-50058-41 - RtBrick-L2TP-Profile

The default L2TP configuration profile assigned in the access configuration profile can be changed in RADIUS access-accept to allow different L2TP configurations based on tunnel endpoints. This is typically used to enable or disable L2TP CSUN updates (RFC5515).

VSA 26-50058-42 - RtBrick-L2TP-Tx-Connect-Speed

VSA 26-50058-43 - RtBrick-L2TP-Rx-Connect-Speed

Those attributes are supported in RADIUS access-accept and CoA requests to set the corresponding L2TP to connect speed values directly. Per default, the L2TP connect speed is derived from the actual data rate upstream/downstream of the configured source (PPPoE-IA or RADIUS) if present or set to port speed which can be overridden with the connect speed attributes. Changes in connect speed via CoA request will trigger the L2TP LAC to send a CSUN request to the LNS if enabled.

Those two attributes are defined as 4-byte integers with speed defined in kbits.

RADIUS Lawful Interception

Lawful interception (LI) refers to the facilities in telecommunications and telephone networks that allow law enforcement agencies (LEA) with court orders or other legal authorization to selectively intercept individual subscribers. The term mediation device (MED) used in this document describes the element which receives and optionally converts the intercepted traffic into the format expected by the law enforcement agencies of the corresponding countries.

All of the following attributes must be present in RADIUS access-accept or CoA request to control lawful interception (LI) via RADIUS. Those attributes are salt encrypted using the algorithm described in RFC 2868 for the Tunnel Password. This encryption algorithm is defined for RADIUS access-accept messages only. To support CoA requests the request authenticator should be replaced with 16 zero bytes which is common industry standard.

The LI action NOOP can be used to obfuscate lawful interception requests (fake requests) to prevent just the presence of those attributes indicating that a subscriber is intercepted. LI requests via RADIUS will show up in the same table as requests via REST or HTTP RPC API (global.access.1.li_request).

Failed LI activations are not signaled via RADIUS to prevent that just the presence of CoA response NAK shows that LI request is not fake (action NOOP).

VSA 26-50058-140 - RtBrick-LI-Action (salt encrypted integer)

Value	Code	Description
NOOP	0	No action / Ignore LI request
ON	1	Start LI / Add LI request
OFF	2	Stop LI / Delete LI request

VSA 26-50058-141 - RtBrick-LI-Identifier (salt encrypted integer)

Device unique lawful interception identifier (LIID) within the range from 1 to 4194303.

VSA 26-50058-142 - RtBrick-LI-Direction (salt encrypted integer)

Value	Code	Description
INGRESS	1	Ingress mirroring only (from subscriber)
EGRESS	2	Egress mirroring only (to subscriber)
BOTH	3	Bidirectional mirroring (from and to subscriber)

VSA 26-50058-143 - RtBrick-LI-MED-Instance (salt encrypted string)

Routing instance through which the mediation device is reachable.

VSA 26-50058-144 - RtBrick-LI-MED-IP (salt encrypted IPv4 address)

IPv4 address of the mediation device.

VSA 26-50058-145 - RtBrick-LI-MED-Port (salt encrypted integer)

UDP port between 49152 and 65535 are set in the mirrored traffic.

RADIUS Framed Routes

The framed-route attributes are used by the RADIUS to install specific routes for a given subscriber. The corresponding next-hop information is automatically derived from the subscriber context. This functionality is typically used to route additional customer networks through the client device (CPE).

Those attributes are supported via RADIUS access-accept only. Each of those attributes may occur multiple times to install various routes. The actual amount of routes supported is limited by the maximum RADIUS packet size only.

The framed-route attributes are defined in RFC 2865 for IPv4 and RFC 3162 for IPv6 but the actual content of those attributes is implementation dependent. RBFS is expecting a string with a prefix followed by prefix length. The prefix can be followed by further routing options like costs, metric or even next-hop which are currently ignored.

Attribute 22 - Framed-Route

```
Framed-Route = "198.51.100.69/24"
```

Attribute 99 - Framed-IPv6-Route

```
Framed-IPv6-Route = "2001:db8:0:90::/32"
```

RADIUS Subscriber ACLs (Filters)

The subscriber ACL attributes are used by the RADIUS server to install a specific ACL, which has already been configured in the BNG, for a subscriber. For information about Subscriber ACL Configuration see 'Subscriber ACLs and HTTP Redirect Service' guide. After the Subscriber ACLs are configured with the name and the rules, use the following RADIUS attributes dynamically for adding or changing those ACLs from RADIUS. These attributes are supported in access-accept and CoA.

The subscriber ACL attributes are used by the RADIUS server to install a specific ACL, which has already been configured in the BNG, for a subscriber. For information about Subscriber ACL Configuration, see the 'Subscriber ACLs and HTTP Redirect Service' guide. After the Subscriber ACLs are configured with the name and the rules, use the following RADIUS attributes to add or change those ACLs from RADIUS. These attributes are supported in access-accept and CoA.

When a CoA is issued with one of those ACL attributes or a service profile, it will always reset all existing filters, even if the new service profile does not define any filters. If there are filters specified within the service profile along with subscriber ACL attributes, these subscriber ACL attributes take precedence. The filters are merged by first applying all the filters from the service profile and then adding or replacing filters using the attributes defined here. The final resulting filter set is then applied to the subscriber.

Filters that were applied first but are not defined in the new set will be deleted. Consequently, adding a single filter via CoA will remove all other existing filters. This approach is designed to provide a more declarative way of defining the desired target state without depending on the existing state.

- ***VSA 26-50058-76 - RtBrick-IPv4-ACL-In***

  ```
  Attach IPv4 subscriber ingress ACL (ACL in upstream direction)
  ```

- ***VSA 26-50058-77 - RtBrick-IPv4-ACL-Out***

  ```
  Attach IPv4 subscriber ingress ACL (ACL in downstream direction)
  ```

- ***VSA 26-50058-78 - RtBrick-IPv6-ACL-In***

  ```
  Attach IPv6 subscriber ingress ACL (ACL in upstream direction)
  ```

- ***VSA 26-50058-79 - RtBrick-IPv6-ACL-Out***

  ```
  Attach IPv6 subscriber egress ACL (ACL in downstream direction)
  ```

Example: RtBrick-IPv4-ACL-In = “ipoe-sub1-http-acl“

To detach any of the preceding ACLs and to delete the ACL associated with a subscriber, null string can be used with the corresponding attribute.

Example: RtBrick-IPv4-ACL-In = ““

RADIUS HTTP Redirect URL

When HTTP Redirect is enabled for a subscriber either through Ascend-Data-Filter or through the subscriber ACLs, HTTP packets from the subscriber get redirected to the URL set by this attribute:

VSA 26-50058-75 - RtBrick-HTTP-Redirect-URL

This attribute is supported on RADIUS access-accept and CoA requests. This attribute allows the overwriting of the URL.

For example:

RtBrick-HTTP-Redirect-URL = “http://portal.rtbrick.com”

To reset the URL:

RtBrick-HTTP-Redirect-URL = “”

RADIUS Sourced DHCP Options

RBFS allows for centralized configuration of DHCP options directly on a RADIUS server, enabling dynamic distribution of these options on a per-subscriber basis. This approach, known as **RADIUS Sourced DHCP Options**, shifts the responsibility of defining and managing DHCP options to the RADIUS server.

Since RBFS performs minimal processing and validation on these options, they are treated as **raw data**, meaning they are forwarded without modification or extensive error checking.

- ***VSA 26-50058-30 - RtBrick-DHCPv4-Client-Options***
- ***VSA 26-50058-32 - RtBrick-DHCPv6-Client-Options***

The client-option attributes allow DHCP options to be included as part of the Access-Accept message. These options are then inserted into outgoing packets directed toward the subscribers DHCP client, regardless of whether RBFS is operating in IPoE relay/proxy mode or server mode.

- ***VSA 26-50058-31 - RtBrick-DHCPv4-Server-Options***
- ***VSA 26-50058-33 - RtBrick-DHCPv6-Server-Options***

Unlike client-option attributes, the server-option attributes are only supported in IPoE relay/proxy mode and are specifically designed to modify packets being forwarded to a DHCP server. This functionality provides additional configuration flexibility when RBFS operates as a relay or proxy between the client and the DHCP server.

For example, server-option attributes can be used to insert specific DHCP options into requests sent to the DHCP server, enabling it to select the appropriate IP address pool based on the provided parameters.

If a RADIUS server includes multiple instances of the same DHCP option within a single Access-Accept message, RBFS will concatenate them into a single data stream. However, the order of the attributes in the packet is crucial for proper reassembly.

For successful concatenation, the option instances must appear consecutively in the RADIUS response. If other attributes are interleaved between multiple instances of **RtBrick-DHCP-Client-Options**, those can’t be concatenated.

FreeRADIUS Example:

```
"02:00:00:00:00:01@ipoe" Cleartext-Password := "ipoe"
    Service-Type = Framed-User,
    Class = IPOE,
    RtBrick-DHCPv4-Client-Options = "0x39020240",
    RtBrick-DHCPv4-Client-Options += "0x3d0701001094000008",
    RtBrick-DHCPv4-Server-Options = "0x39020240",
    RtBrick-DHCPv6-Server-Options = "0x001100100000c38a00020008435352 2D56365044",
    RtBrick-DHCPv6-Client-Options = "0x001100100000c38a00010008435352 2D56365044"
```

Example Option:

```
0x39020240

  39    Code      // DHCP Max Msg Size
  02    Length    // 2 Byte
  0240  Data      // 576 Byte
```

Since options are added as raw data, they must be in network byte order.

DHCPv6 and DHCPv4 operate differently when it comes to the inclusion of client and server options within DHCP messages.

In DHCPv6, server options are incorporated into the outermost DHCPv6 relay header. This specific header is added by the relay agent (RBFS). On the other hand, client options are added to the innermost DHCPv6 message, but this only occurs if the message type is either an advertisement (type 2) or a reply (type 7).

Conversely, in DHCPv4, the placement of client and server options is based on specific message types. Client options are exclusively added to offer (type 2) and ack (type 5) messages. In contrast, server options are only included in discovery (type 1) and request (type 3) messages.

The following DHCP options are not permitted and will be filtered out if attempted to be added via RADIUS:

DHCPv4:

- **0:** Pad Option
- **51:** IP Address Lease Time
- **52:** Option Overload
- **53:** DHCP Message Type
- **54:** Server Identifier
- **55:** Parameter Request List
- **255:** End

DHCPv6:

- **1:** OPTION_CLIENTID
- **2:** OPTION_SERVERID
- **3:** OPTION_IA_NA
- **6:** OPTION_ORO
- **8:** OPTION_ELAPSED_TIME
- **25:** OPTION_IA_PD

The DHCP options will add received options even if they are already present in the original packet unless they are filtered out as not allowed or invalid. For the DHCPv4 server options listed below, the original option is removed before the same option is added via the DHCPv4 server options to prevent duplicate options.

Other options may still become duplicates.

- Client-Identifier (61)
- Relay-Agent (82)

RADIUS Terminate Codes

VSA 26-50058-27 - RtBrick-Terminate-Code

The RtBrick-Terminate-Code is sent along with the standard Acct-Terminate-Cause (attribute 49). The list below shows the RtBrick termination codes with the mapping to the standard cause.

RtBrick-Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
0	10 (NAS Request)	NA
1	10 (NAS Request)	Subscriber Management Unknown Error
2	9 (NAS Error)	SubscriberD Internal Error
3	9 (NAS Error)	PPPoED Internal Error
4	9 (NAS Error)	PPPoED Object Deleted *This code is used if an PPPoE session object is deleted which is an indication for a crash or some other issues in PPPoE daemon.*
5	9 (NAS Error)	L2TPD Internal Error
6	9 (NAS Error)	L2TPD Object Deleted *This code is used if an L2TP session object is deleted which is an indication for a crash or some other issues in L2TP daemon.*
7	9 (NAS Error)	AAA Profile Not Found
8	9 (NAS Error)	RADIUS Profile Not Found
9	9 (NAS Error)	Authentication Type Missing

RtBrick-Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
10	9 (NAS Error)	Authentication Order Missing or Invalid
11	10 (NAS Request)	Authentication Failure
12	10 (NAS Request)	Local Authentication Failed
13	10 (NAS Request)	Accounting-Request-On Wait
14	9 (NAS Error)	No RADIUS Authentication Server Configured
15	10 (NAS Request)	RADIUS Authentication Failed
16	17 (User Error)	Authentication Rejected
17	17 (User Error)	Local Authentication Rejected
18	17 (User Error)	RADIUS Authentication Rejected
19	5 (Session Timeout)	Session Timeout
20	4 (Idle Timeout)	Idle Timeout
21	6 (Admin Reset)	Clear Session
22	6 (Admin Reset)	RADIUS CoA Disconnect
23	9 (NAS Error)	PPPoE Unknown Error
24	1 (User Request)	PPPoE PADT Received
25	9 (NAS Error)	PPPoE LCP Error
26	10 (NAS Request)	PPPoE LCP Generic Error
27	1 (User Request)	PPPoE LCP Terminate Request Received
28	10 (NAS Request)	PPPoE LCP Maximum Reject / NAK Exceeded
29	10 (NAS Request)	PPPoE LCP Negotiation Failed
30	10 (NAS Request)	PPPoE LCP Configuration-Request Exceeded
31	10 (NAS Request)	PPPoE LCP Echo-Request Timeout Exceeded

RtBrick-Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
32	9 (NAS Error)	PPPoE PAP Error
33	9 (NAS Error)	PPPoE CHAP Error
34	9 (NAS Error)	PPPoE IPCP Error
35	9 (NAS Error)	PPPoE IP6CP Error
36	9 (NAS Error)	PPPoE NCP Initialization Failed
37	10 (NAS Request)	PPPoE NCP Down
38	6 (Admin Reset)	PPPoE Clear Session
39	10 (NAS Request)	PPPoE Upper Layer Down
40	8 (Port Error)	PPPoE Interface Down
41	10 (NAS Request)	PPPoE Configuration Deleted
42	9 (NAS Error)	PPPoE Access Configuration Profile Not Found
43	9 (NAS Error)	L2TP Unknown Error
44	10 (NAS Request)	L2TP Tunnel Down
45	10 (NAS Request)	L2TP Tunnel Dead
46	10 (NAS Request)	L2TP Tunnel Deleted
47	10 (NAS Request)	L2TP No Tunnel Available
48	10 (NAS Request)	L2TP CDN Request
49	9 (NAS Error)	L2TP Profile Not Found
50	9 (NAS Error)	L2TP Access Configuration Profile Not Found
51	9 (NAS Error)	L2TP No Tunnel Pool Error
52	9 (NAS Error)	L2TP Local Tunnel Pool Error
53	9 (NAS Error)	L2TP RADIUS Tunnel Pool Error
54	6 (Admin Reset)	L2TP Clear Session
55	9 (NAS Error)	Access Configuration Profile Not Found

RtBrick-Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
56	9 (NAS Error)	Local IPv4 Address Pool Not Found
57	10 (NAS Request)	Local IPv4 Address Pool Exhausted
58	9 (NAS Error)	Local IPv6 Prefix Pool Not Found
59	10 (NAS Request)	Local IPv6 Prefix Pool Exhausted
60	9 (NAS Error)	Local IPv6 Delegated Prefix Pool Not Found
61	10 (NAS Request)	Local IPv6 Delegated Prefix Pool Exhausted
62	10 (NAS Request)	PPPoE CHAP Response Timeout
63	10 (NAS Request)	L2TP Session Deleted
64	10 (NAS Request)	Duplicate IPv4 address detected
65	10 (NAS Request)	Duplicate IPv6 prefix detected
66	10 (NAS Request)	Duplicate delegated IPv6 prefix detected
67	9 (NAS Error)	Routing instance not found
68	6 (Admin Reset)	Clear Session Force
69	6 (Admin Reset)	Test AAA Request Object Deleted
70	9 (NAS Error)	IPoED Object Deleted *This code is used if an IPoE subscriber object is deleted which is an indication for a crash or some other issues in IPoE daemon.*
71	9 (NAS Error)	IPoED Internal Error

RtBrick-Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
72	9 (NAS Error)	IPoE Unknown Error
73	10 (NAS Request)	IPoE Upper Layer Down
74	10 (NAS Request)	IPoE Lower Layer Down
75	10 (NAS Request)	IPoE Interface Down
76	10 (NAS Request)	IPoE Configuration Deleted
77	10 (NAS Request)	SubscriberD Request
78	10 (NAS Request)	L2BSA Service Deleted
79	10 (NAS Request)	L2BSA Service Added
80	6 (Admin Reset)	IPoE Clear Subscriber
81	6 (Admin Reset)	PPPoE Clear Session Force
82	9 (NAS Error)	IPoE Invalid Gateway IFL
83	9 (NAS Error)	FWD Invalid
84	9 (NAS Error)	FWD Failed
85	9 (NAS Error)	FWD Deleted
86	10 (NAS Request)	Instance Deleted
87	10 (NAS Request)	Redundancy Switchover
88	10 (NAS Request)	Redundancy Active Node Termination
89	10 (NAS Request)	Redundancy Active Node Down
90	9 (NAS Error)	Redundancy Invalid State
91	9 (NAS Error)	Redundancy Invalid State for Sync
92	9 (NAS Error)	Redundancy Validation Failed
93	10 (NAS Request)	Redundancy Config Deleted
94	9 (NAS Error)	Redundancy Stale Timer Expiry
95	9 (NAS Error)	IPv4 In ACL Validation Failed
96	9 (NAS Error)	IPv4 Out ACL Validation Failed
97	9 (NAS Error)	IPv6 In ACL Validation Failed

RtBrick-Terminate-Code	Acct-Terminate-Cause (RADIUS Attribute 49)	Description
98	9 (NAS Error)	IPv6 Out ACL Validation Failed
99	9 (NAS Error)	IP Address Allocation Failed
100	9 (NAS Error)	No IP Address
101	6 (Admin Reset)	Subscriber Deleted
102	10 (NAS Request)	Redundancy No Interface To Assign *This code is used if a forwarding subscriber object is deleted which is an indication of a crash or some other issues in the forwarding infrastructure.*

It is recommended to raise operational alarms for every termination cause received with a value of 9 (NAS Error) for further investigations.

RADIUS CoA Error Handling

CoA requests should be retried a few times in the unlikely event of CoA NAK with error-cause (RFC 5176 attribute 101) other than session-context-not-found (503) or missing-attribute (402). There is no rollback of failed CoA requests to the former state but retrying the same request is supported because changes are implemented in a more declarative way.

5.2.4. RADIUS Accounting

RADIUS accounting servers handle accounting records for subscribers. The subscriber daemon transmits RADIUS Accounting-Start, Interim and Stop messages to these servers. Accounting is the process of tracking subscriber activity and network resource usage in a subscriber session. This includes the session time called time accounting and the number of packets and bytes transmitted during the session called volume accounting.

A RADIUS Acct-Status-Type attribute is used by the RADIUS client (subscriber daemon) to mark the start of accounting (for example, upon booting) by specifying Accounting-On and to mark the end of accounting (for example, just before a scheduled reboot) by specifying Accounting-Off. This message is often used by RADIUS servers to automatically close/terminate all open accounting records/sessions for the corresponding client and therefore must not be sent to servers belonging to a profile which was already used/started for accounting.

Per default, the assumption is that all servers referenced by a RADIUS profile share the same states and therefore accounting-on must be only sent to one of those before the first accounting-start is sent.

RADIUS Accounting-On/Off messages are disabled by default and can be optionally enabled in the RADIUS profile configuration. There is also an additional configuration option to optionally wait for Accounting-On response which prevents any new session until accounting has started meaning that Accounting-On response is received.

Accounting-Off is currently not implemented!

RADIUS accounting requests are often used for billing and therefore should be able to store and retry over a longer period (common up to 24 hours or more) which can be optionally enabled in the RADIUS profile configuration using the accounting backup configuration option.

Time Accounting

All accounting relevant timestamps are retrieved using the UNIX system call clock_gettime internally stored in the datastore as 64 bit microseconds timestamp with the lower order 32 bit representing the seconds since January 1, 1970 00:00 UTC which is also known as epoch or unix timestamp and the higher order 32 bit representing the microseconds which are per definition less than 1.000.000 (one second). The seconds part is counted in full seconds which is similar to always rounded down.

The RADIUS attribute event-timestamp (type 55) defined in RFC2869 is included in each RADIUS Accounting-Request packet to record the time that this event occurred in seconds since January 1, 1970 00:00 UTC. A max allowed deviation of 500 milliseconds of a timestamp in seconds requires mathematical rounding of the internal 64 bit microseconds timestamps to the 32 bit RADIUS event-timestamp in seconds. This means to round down if the microseconds part of the timestamp is less than 0.5 seconds and rounded up if equal or greater than 0.5 seconds.

Start Timestamps

In RBFS the timestamp indicating the PPPoE session start will be generated after successful negotiation of at least one of the PPP network control protocols (IPCP for IPv4 or IP6CP for IPv6). In case of L2TP tunneled sessions (LAC), the successful L2TP session setup after sending ICCN is used as a start timestamp. In both cases this is also the trigger for the RADIUS accounting-request start where this timestamp is used as RADIUS event-timestamp.

Stop Timestamps

The timestamp indicating the session stop will be generated more or less immediately after termination request (e.g. timeout, user request, nas request, …).

The RADIUS attribute Acct-Session-Time (type 46) defined in RFC2866 indicates how many seconds the user has received service and can only be present in Accounting-Request records where the Acct-Status-Type is set to Interim or Stop. This time will be calculated based on the internal session start and stop time in microseconds and mathematical rounded to seconds. For RADIUS accounting interim requests the current event time is used instead of stop time.

RFC and draft compliance are partial except as specified.

Volume Accounting

Based on §45g of the German Telecommunications Act (TKG), the maximum deviation for accounting relevant counters must be less than 1% per billing period which is typically one month.

The whole RBFS architecture is based on an event-based publish and subscribe model using the BDS infrastructure. Each process publishes all information and states to BDS and interested processes subscribe to that information. The whole inter process communications (IPC) is replaced by BDS table operations which allows to build a hyper scaled distributed software system. This means related to accounting that each counter is requested from data plane and updated in BDS unsolicited based on configurable intervals typically set between 5 to 30 seconds. This applies to any type of counters like interface counters or QoS statistics. This means that CLI commands or API requests will return the last updated counter and not the counter of the time of request. Therefore each BDS object contains a timestamp indicating the time of the last counter update which allows to use these counters with the required accuracy.

Subscriber volume accounting is based on multiple sources like logical interface

(LIF), class/queue-, policer- and control-traffic statistics.

All those counters are layer 2 (L2) counters and some of them may reset through configuration changes like QoS counters after applying a new QoS profile. Therefore the subscriber management application daemon is doing some post processing of all counters to ensure that no accounting information is lost and to calculate the final accounting values. This includes also optional configurable counter adjustments.

There is a BDS database object in the table **local.access.subscriber.accounting** created for each subscriber which stores all counters and attributes required to calculate the actual volume counters. This object is created together with the subscriber and automatically deleted if the actual subscriber object is removed.

The volume accounting counters must not reset if the actual hardware counter is deleted or has changed/reset to zero. Therefore instead of using the source counter directly, the delta since the last interval is calculated and added to the final counter.

The usage of callback functions to counter delete and change events ensures that no accounting relevant information will be lost through reconfiguration of subscriber sessions.

The function which removes the subscriber counters from the data plane will receive the final counters before removal and update them into BDS counter objects before this is deleted.

Interims Volume Counters

RADIUS accounting interim requests will also use the last updated counters with current time as event-timestamp.

Stop Volume Counters

The session termination might be triggered by REST API, CLI (clear subscriber …) RADIUS CoA disconnect request, session timeout, idle timeout or client request (PADT). The corresponding RADIUS accounting stop requests will be delayed to wait for the final counter update but uses the timestamp of the terminate request.

Interims Accounting

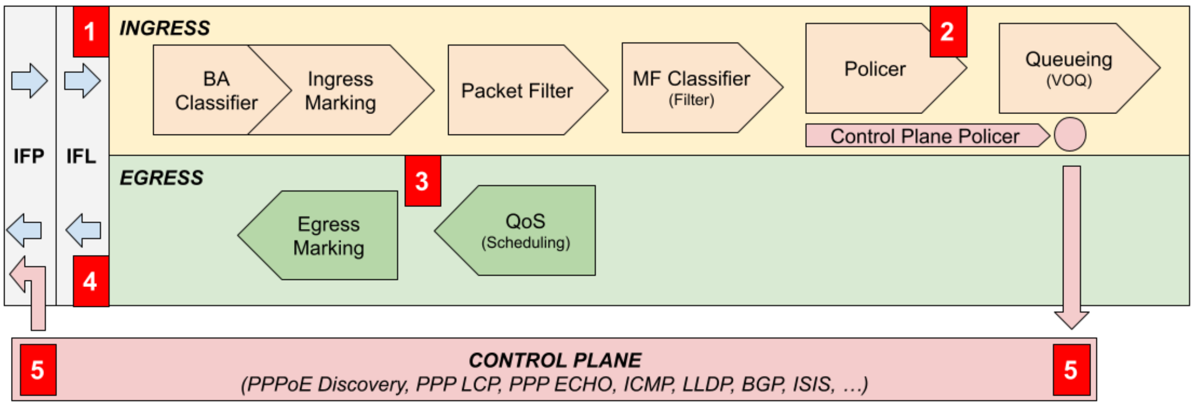
To receive RADIUS counters in fast intervals the corresponding session interim accounting interval can be set depending on the actual needs to any value between one second and multiple hours or days (recommended is at least 30 seconds). This interval can be also updated via CoA request using the Acct-Interim-Interval attribute which triggers an interim accounting immediately and uses the new interval from now onwards. This can be also used to request interim accounting requests on demand. Sending a CoA request with the applied interval triggers an accounting request but the original interval is not changed.

5.2.5. RADIUS Counters

The packets and byte counters of each session, traffic class (class 0 - 7) as well as policer counters (level 1 to 4) are supported via RADIUS accounting interims and stop messages. Those counters are layer two counters per default which can be adjusted using correction values and factors from the AAA configuration profile.

Subscriber accounting is based on multiple sources like LIF-, class/queue-, policer- and control-traffic statistics which are described below.

- The InLIF (Incoming Logical Interface) stats count all traffic received on the logical interface including control traffic are traffic dropped by ingress policer.
- The policer stats count all traffic accepted by policer (color green) per level (1-4). Ingress control traffic will be hit by a separate control plane policer and therefore not counted in the session policer stats.
- The class/queue stats count all egress traffic except control traffic which is directly sent to the IFP.
- The OutLIF (Outgoing Logical Interface) stats count all traffic sent on the logical interface except control traffic which is directly sent to the IFP.
- The control stats count all traffic received and sent from or to the control plane for a given subscriber (counted in PPPoE daemon for PPPoE sessions).



Because counter updates are not atomic operations, the sum of all class counters might be different from the session counters.

The counted bytes per packet can be adjusted as described in chapter Configuring Accounting Adjustments of the Subscriber Management Configuration Guide.

All subscriber accounting attributes and counters are stored without adjustment (L2) in the table **local.access.subscriber.accounting** and remain until the subscriber session is finally removed after response to RADIUS accounting stop.

The command show subscriber <id> accounting displays the adjusted subscriber accounting information. It is also possible to display the values before adjustment using show subscriber <id> accounting origin.

```
supervisor@rtbrick: op> show subscriber 72339069014638637 accounting
Subscriber-Id: 72339069014638637
    IFL: ppp-0/0/1/72339069014638637
    Start Timestamp: Wed Feb 24 09:06:47 GMT +0000 2021
    Idle Timestamp: Wed Feb 24 09:06:47 GMT +0000 2021
    Session-Timeout: 86400 seconds
    Idle-Timeout: 7200 seconds
    Session Statistics:
        Ingress: 0 packets 0 bytes
        Egress: 0 packets 0 bytes
    LIF Statistics:
        Ingress: 14 packets 896 bytes
        Egress: 0 packets 0 bytes
    Egress Class (Queue) Statistics:
        class-0: 0 packets 0 bytes
        class-1: 0 packets 0 bytes
        class-2: 0 packets 0 bytes
        class-3: 0 packets 0 bytes
        class-4: 0 packets 0 bytes
        class-5: 0 packets 0 bytes
        class-6: 0 packets 0 bytes
        class-7: 0 packets 0 bytes
    Ingress Policer Statistics:
        Level 1: 0 packets 0 bytes
        Level 2: 0 packets 0 bytes
        Level 3: 0 packets 0 bytes
        Level 4: 0 packets 0 bytes
```

Session Counters

Per default, the session counters are calculated using the LIF statistics which include all traffic received on the logical interface (InLIF) and all traffic sent on the logical interface (OutLIF) except control traffic which is directly sent to the IFP. Ingress traffic sent to CPU or transit traffic dropped by policer is still counted whereas egress traffic dropped by QoS is not counted.

Alternatively, it is possible to use the sum of all policer counters for ingress session counters which can be changed in the AAA configuration profile by setting the accounting ingress source to POLICER (default is LIF) to include only accepted transit traffic.

Using LIF in egress and POLICER in ingress results in a symmetric behavior where only accepted transit traffic is counted.

Following the list of the RtBrick class counter attributes.

Attribute	Name	Description
42	Acct-Input-Octets	Incoming session bytes (uint32)
43	Acct-Output-Octets	Outgoing session bytes (uint32)
47	Acct-Input-Packets	Incoming session packets (uint32)
48	Acct-Output-Packets	Outgoing session packets (uint32)
52	Acct-Input-Gigawords	Acct-Input-Octets overflow counter (uint32)
53	Acct-Output-Gigawords	Acct-Output-Octets overflow counter (uint32)

The internal 64 bit counters are split over the standard 32 bit octet and gigaword counters by using the most significant 32 bit as gigawords and the least significant 32 bits as octets.

32 Bit Acct-Input/Output-Gigawords	32 Bit Acct-Input/Output-Octets
Internal 64 Bit Counter	

Class Counters

The outgoing class counters are filled by queue counters which ensures that only traffic leaving the device is counted here.

Following the list of the RtBrick class counter attributes.

Attribute	Name	Description
26-50058-81	RtBrick-Class-0-Packets-Out	Outgoing Class-0 packets (uint64)
26-50058-82	RtBrick-Class-0-Bytes-Out	Outgoing Class-0 bytes (uint64)
26-50058-83	RtBrick-Class-1-Packets-Out	Outgoing Class-1 packets (uint64)
26-50058-84	RtBrick-Class-1-Bytes-Out	Outgoing Class-1 bytes (uint64)
26-50058-85	RtBrick-Class-2-Packets-Out	Outgoing Class-2 packets (uint64)
26-50058-86	RtBrick-Class-2-Bytes-Out	Outgoing Class-2 bytes (uint64)
26-50058-87	RtBrick-Class-3-Packets-Out	Outgoing Class-3 packets (uint64)
26-50058-88	RtBrick-Class-3-Bytes-Out	Outgoing Class-3 bytes (uint64)
26-50058-98	RtBrick-Class-4-Packets-Out	Outgoing Class-4 packets (uint64)
26-50058-90	RtBrick-Class-4-Bytes-Out	Outgoing Class-4 bytes (uint64)
26-50058-91	RtBrick-Class-5-Packets-Out	Outgoing Class-5 packets (uint64)
26-50058-92	RtBrick-Class-5-Bytes-Out	Outgoing Class-5 bytes (uint64)
26-50058-93	RtBrick-Class-6-Packets-Out	Outgoing Class-6 packets (uint64)

Attribute	Name	Description
26-50058-94	RtBrick-Class-6-Bytes-Out	Outgoing Class-6 bytes (uint64)
26-50058-95	RtBrick-Class-7-Packets-Out	Outgoing Class-7 packets (uint64)
26-50058-96	RtBrick-Class-7-Bytes-Out	Outgoing Class-7 bytes (uint64)

Those attributes will be encoded as subattributes (RFC2865) similar to broadband forum (BBF) line attributes to reduce the size of the RADIUS message. Counters with a zero value will be also excluded from the packet.

```
▼ AVP: t=Vendor-Specific(26) l=246 vnd=RtBrick Inc.(50058)
    Type: 26
    Length: 246
    Vendor ID: RtBrick Inc. (50058)
  ▶ VSA: t=RtBrick-Class-0-Packets-Out(81) l=10 val=000000000000000a
  ▶ VSA: t=RtBrick-Class-0-Bytes-Out(82) l=10 val=00000000000003e8
  ▶ VSA: t=RtBrick-Class-1-Packets-Out(83) l=10 val=000000000000000b
  ▶ VSA: t=RtBrick-Class-1-Bytes-Out(84) l=10 val=00000000000003e9
  ▶ VSA: t=RtBrick-Class-2-Packets-Out(85) l=10 val=000000000000000c
  ▶ VSA: t=RtBrick-Class-2-Bytes-Out(86) l=10 val=00000000000003ea
  ▶ VSA: t=RtBrick-Class-3-Packets-Out(87) l=10 val=000000000000000d
  ▶ VSA: t=RtBrick-Class-3-Bytes-Out(88) l=10 val=00000000000003eb
  ▶ VSA: t=RtBrick-Class-4-Packets-Out(89) l=10 val=000000000000000e
  ▶ VSA: t=RtBrick-Class-4-Bytes-Out(90) l=10 val=00000000000003ec
  ▶ VSA: t=RtBrick-Class-5-Packets-Out(91) l=10 val=000000000000000f
  ▶ VSA: t=RtBrick-Class-5-Bytes-Out(92) l=10 val=00000000000003ed
  ▶ VSA: t=RtBrick-Class-6-Packets-Out(93) l=10 val=0000000000000010
  ▶ VSA: t=RtBrick-Class-6-Bytes-Out(94) l=10 val=00000000000003ee
  ▶ VSA: t=RtBrick-Class-7-Packets-Out(95) l=10 val=0000000000000011
  ▶ VSA: t=RtBrick-Class-7-Bytes-Out(96) l=10 val=00000000000003ef
  ▶ VSA: t=RtBrick-Policer-L1-Packets-In(97) l=10 val=0000000000000015
  ▶ VSA: t=RtBrick-Policer-L1-Bytes-In(98) l=10 val=00000000000007d1
  ▶ VSA: t=RtBrick-Policer-L2-Packets-In(99) l=10 val=0000000000000016
  ▶ VSA: t=RtBrick-Policer-L2-Bytes-In(100) l=10 val=00000000000007d2
  ▶ VSA: t=RtBrick-Policer-L3-Packets-In(101) l=10 val=0000000000000017
  ▶ VSA: t=RtBrick-Policer-L3-Bytes-In(102) l=10 val=00000000000007d3
  ▶ VSA: t=RtBrick-Policer-L4-Packets-In(103) l=10 val=0000000000000018
  ▶ VSA: t=RtBrick-Policer-L4-Bytes-In(104) l=10 val=00000000000007d4
```

Policer Counters

The incoming policer attributes count all traffic accepted (colored green) by ingress policers with dedicated packet and byte values per level.

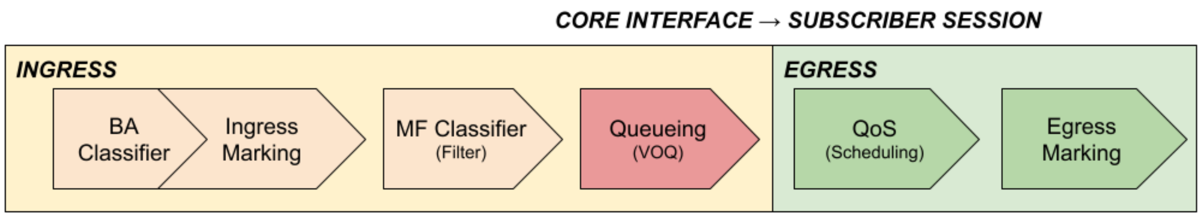
Following the list of the RtBrick policer counter attributes.

Attribute	Name	Description
26-50058-97	RtBrick-Policer-L1-Packets-In	Incoming Policer L1 accepted packets (uint64)
26-50058-98	RtBrick-Policer-L1-Bytes-In	Incoming Policer L1 accepted bytes (uint64)
26-50058-99	RtBrick-Policer-L2-Packets-In	Incoming Policer L2 accepted packets (uint64)
26-50058-100	RtBrick-Policer-L2-Bytes-In	Incoming Policer L2 accepted bytes (uint64)
26-50058-101	RtBrick-Policer-L3-Packets-In	Incoming Policer L3 accepted packets (uint64)
26-50058-102	RtBrick-Policer-L3-Bytes-In	Incoming Policer L3 accepted bytes (uint64)
26-50058-103	RtBrick-Policer-L4-Packets-In	Incoming Policer L4 accepted packets (uint64)
26-50058-104	RtBrick-Policer-L4-Bytes-In	Incoming Policer L4 accepted bytes (uint64)

Those attributes will be encoded as subattributes (RFC2865) similar to broadband forum (BBF) line attributes to reduce the size of the RADIUS message. Counters with a zero value will be also excluded from the packet.

Service Classification and Accounting

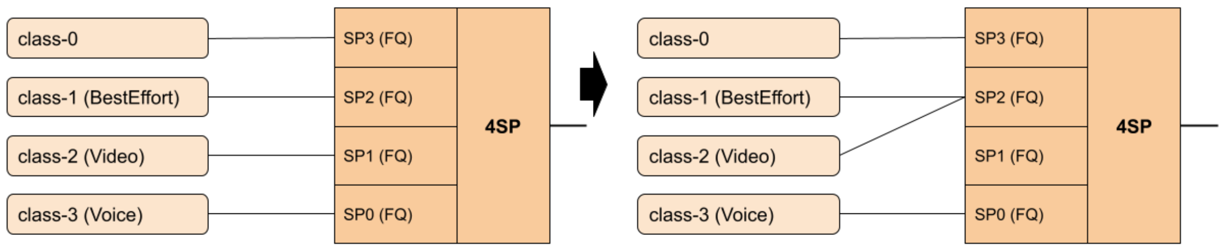
The underlying virtual output queueing (VOQ) is a common and efficient architecture for high performance switches but brings some major limitations. One obvious limitation is that queues must be allocated in ingress and canΓÇÖt be changed in egress as shown in the picture below.



Service classification and accounting is based on standard behavior aggregate (BA) and multifield classifier (MF) bound global or to ingress interfaces matching on some protocol fields assigning a traffic class. Downstream traffic (to subscriber) is counted per subscriber and class using queue counters to count only traffic not dropped by QoS. Upstream traffic (from subscribers) is countered per subscriber and policer level.

Some service providers would like to control if premium traffic is handled differently on a per subscriber basis which can be also achieved with VOQ as explained below.

Let us assume a product which handles video streaming services with higher priority or excludes those traffic from actual data volume. With VOQ the classification as Video is done on the ingress core interface equally for all subscribers regardless of the booked product. Now instead of changing the queue per subscriber in egress, the behavior of the queue can be changed to behave equally to BestEffort which is supported in egress per subscriber.



The corresponding Video traffic is now threaded equally to BestEffort but still counted separately. For accounting purposes the traffic of BestEffort and Video can be simply added in the billing infrastructure to get the actual amount of BestEffort traffic.

In upstream, the expected class and policer level can be assigned per subscriber because BA and MF classifiers are bound to the PPPoE session in ingress.

5.2.6. RADIUS Accounting Adjustment Configuration

The accounting adjustment allows to do some basic counter adjustment for RADIUS interims and stop accounting request messages using the following parameters from the AAA configuration profile (**global.access.aaa.profile.config**).

This counter adjustment allows normalizing counters with different encapsulations (double tagged, untagged, …) to L3 counters for example.

Parameter	Type	Description
accounting_ingress_source	uint8	Source of session ingress counter (1: LIF or 3: POLICER) *Default: LIF*
accounting_egress_source	uint8	Source of session egress counter (1: LIF or 2: CLASS) *Default: LIF*
ingress_byte_adjustment_value	float	Adjust ingress LIF counters by +/-N bytes per packet *Default: 0.00*
ingress_byte_adjustment_factor	float	Adjust ingress LIF counters by factor (executed after adjustment value) *Default: 1.00*
egress_byte_adjustment_value	float	Adjust egress LIF counters by +/-N bytes per packet *Default: 0.00*
egress_byte_adjustment_factor	float	Adjust egress LIF counters by factor (executed after adjustment value) *Default: 1.00*
accounting_egress_class_byte_adjustment_value	float	Adjust egress CLASS counters by +/-N bytes per packet *Default: 0.00*
accounting_egress_class_byte_adjustment_factor	float	Adjust egress POLICER counters by factor (executed after adjustment value) *Default: 1.00*

Parameter	Type	Description
accounting_ingress_policer_byte_adjustment_value	float	Adjust ingress POLICER counters by +/-N bytes per packet *Default: 0.00*
accounting_ingress_policer_byte_adjustment_factor	float	Adjust ingress LIF counters by factor (executed after adjustment value) *Default: 1.00*

The byte adjustment value supports positive and negative values like -20.0 or 20.0. Provided decimal digits in the adjustment values are ignored.

The byte adjustment factors support positive values and only the first two decimal digits are used like 0.98 (-2%) or 1.02 (+2%).

5.2.7. RADIUS Redundancy

It is possible to configure multiple RADIUS authentication and accounting servers for redundancy and or load-balancing.

The following two algorithms are supported:

- **DIRECT (default):** Requests are sent to the server following the one where the last request was sent. If the subscriber daemon receives no response from the server, requests are sent to the next server and so on.
- **ROUND-ROBIN:** Requests are sent to the server following the one where the last request was sent. If the subscriber daemon router receives no response from the server, requests are sent to the next server and so on.

5.2.8. Test AAA

The test AAA subscriber feature allows operators to verify and test authentication and accounting (e.g. local or RADIUS) by emulating a subscriber without the need for external clients to be connected. This is a commonly used feature used during troubleshooting or to validate the RADIUS configuration.

Test subscribers will be created by adding a request object into the test aaa request table (**global.subscriber.1.test.aaa.request**) via API or CLI. This request object includes beside username and password also a spoofed interface, outer-vlan, inner-vlan and MAC address required to build corresponding attributes like NAS port identifiers or the vendor specific RtBrick-Access-Stack. It is also possible to add an Agent-Circuit-Id or Agent-Remote-Id to test line based authentication.

This new object will first trigger a validation plugin to reject invalid requests (e.g. subscriber-id out of range, missing mandatory attributes, …). The add callback for this object will trigger the creation of the actual subscriber. Deleting the request object will trigger the termination of the subscriber. If this subscriber terminates for other reasons like CoA or CLI, the subscriber remains in a terminated state until the request object is deleted.

Compared to real subscribers, test subscribers will not trigger any forwarding state. Therefore no object will be created in the following tables for those subscribers:

- global.access.1.subscriber.ifl

But the following tables will be populated for operators to check dynamic QoS, filters or IGMP:

- global.access.1.subscriber.qos
- global.access.1.subscriber.qos.shaper
- global.access.1.subscriber.filter
- global.access.1.subscriber.igmp.service

Those entries are ignored internally as there is no matching subscriber interface (IFL) (global.access.1.subscriber.ifl). The only exception here is the filter table meaning that received filters (ADF) learned from RADIUS are installed in the ACL table and applied with the IP addresses of the test subscribers.

Test AAA Subscriber-Id:

In RBFS each subscriber is uniquely identified by subscriber-id which is a 64 bit number allocated in a distributed fashion by the application creating the subscriber (for example, pppoed for pppoe sessions). This uniqueness is achieved with the following format.

The first 8 bits are used to identify the application:

- 0x00 subscriberd
- 0x01 pppoed
- 0x02 l2tpd

The next 8 bits are used to identify the application sharding instance allocated (for example, pppoed.1, pppoed.2, and so on).

The remaining 48 bits are used to uniquely identify the subscriber, which theoretically allows up to 281474976710656 subscribers per application instance.

An emulated test subscriber needs also a subscriber-id.

```
-----
* 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

* +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

* | app-id (0) | app-instance | reserved |

* +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

* | subscriber

* +-+-+-+-+

-----
```

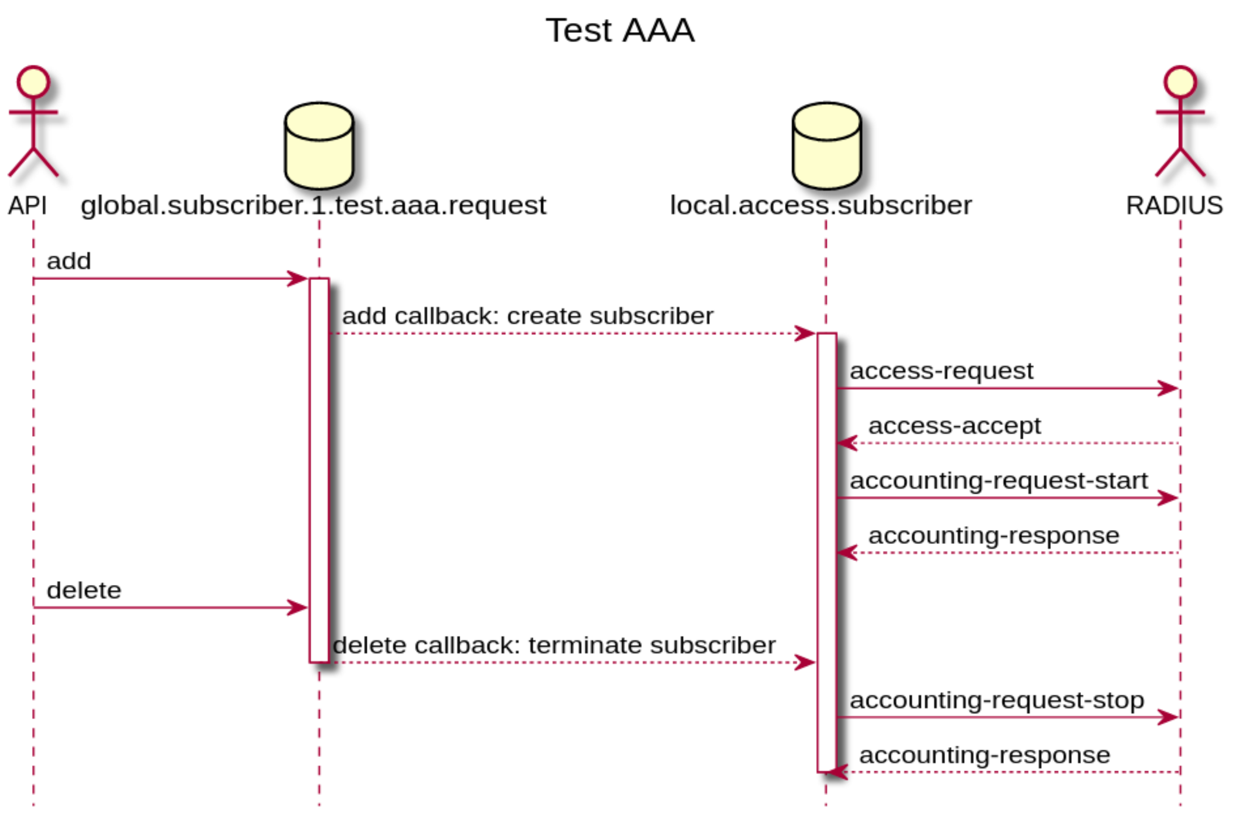
Assuming subscriber daemon instance 1 (subscriberd.1), the valid range for test subscriber identifiers is between 281474976710656 and 281479271677951.

```
Min Subscriber-Id: 281474976710656
0000 0000 0000 0001 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Max Subscriber-Id: 281479271677951
0000 0000 0000 0001 0000 0000 0000 0000 1111 1111 1111 1111 1111 1111 1111 1111
```

The subscriber identifier will be allocated manually by operators from this range which has the advantage to easier identify the subscriber created by request.

Test AAA RADIUS Flow:



Test AAA Attributes

Attribute	Mandatory	API	CLI	Default
Subscriber-Id	Yes	subscriber_id	subscriber-id	
Access Profile	Yes	access_profile_name	access-profile	
AAA Profile	Yes	aaa_profile_name	aaa-profile	
Username	No	user_name	username	test
Password	No	password	password	test
Agent-Circuit-id	No	aci	aci	
Agent-Remote-id	No	ari	ari	
Interface (IFP)	*	interface_name	interface	ifp-0/0/0
Outer VLAN	No	outer_vlan	outer-vlan	0
Inner VLAN	No	inner_vlan	inner-vlan	0

Attribute	Mandatory	API	CLI	Default
MAC Address	No	client_mac	**	00:00:00:00:00:00

** The interface is mandatory via CLI but optional via API.*

*** The client MAC address is currently supported via API only.*

Test AAA via CLI

In CLI, strings must not contain the “” character and therefore this is not permitted for username, aci or ari set via CLI. This limitation is valid for CLI only as subscribers created via API are allowed to use the “” character in the mentioned strings.

Add Test Subscriber:

```
supervisor@leaf1: op> test subscriber aaa request 281474976710656 pppoe-dual aaa-
default ifl-0/0/1 username user1@rtbrick.com aci "0.0.0.0/0.0.0.0 eth 0/0" ari
DEU.RTBRICK.01
```

List All Test Subscribers:

```
supervisor@leaf1: op> show subscriber filter type Test
Subscriber-Id          Interface        VLAN      State             Username
281474976710656        ifp-0/1/23       0:0       ESTABLISHED
user1@rtbrick.com
```

Delete Test Subscriber:

```
supervisor@leaf1: op> test subscriber aaa delete 281474976710656
```

Test AAA via API

Add Test Subscriber:

```
{{rbfs_url}}/api/v1/rbfs/elements/{\{element}}/services/subscriberd.1/proxy/bds/ob
ject/add
```

```
{
   "table": {
       "table_name": "global.access.1.test.aaa.request"
   },
   "objects": [
       {
           "attribute": {
               "subscriber_id": 281474976710658,
               "aaa_profile_name": "aaa-default",
               "access_profile_name": "pppoe-dual",
               "user_name": "user@rtbrick.com",
               "ari": "DEU.RTBRICK.01",
               "aci": "0.0.0.0/0.0.0.0 eth 0/0"
           }
       }
   ]
}
```

Delete Test Subscriber:

```
{{rbfs_url}}/api/v1/rbfs/elements/{\{element}}/services/subscriberd.1/proxy/bds/ob
ject/delete
```

```
{
    "table": {
        "table_name": "global.access.1.test.aaa.request"
    },
    "objects": [
        {
            "attribute": {
                "subscriber_id": 281474976710658
            }
        }
    ]
}
```

List All Test Subscribers:

```
{{rbfs_url}}/api/v1/rbfs/elements/{\{element}}/services/subscriberd.1/proxy/bds/ta
ble/walk
```

```
{
    "table": {
        "table_name": "global.access.1.test.aaa.request"
    }
}
```

5.2.9. Appendix A - RADIUS Dictionary

Following the RtBrick RADIUS dictionary in the well known FreeRADIUS format.

```
# This dictionary applies to RtBrick Full Stack (RBFS)
# https://www.rtbrick.com/
#
VENDOR          RtBrick 50058

BEGIN-VENDOR    RtBrick

ATTRIBUTE       RtBrick-Access-Hostname             21  string
ATTRIBUTE       RtBrick-Access-Port                 22  string
ATTRIBUTE       RtBrick-Access-Stack                23  string
ATTRIBUTE       RtBrick-Access-MAC-Address          24  string

ATTRIBUTE       RtBrick-Subscriber-Id               25  integer64
ATTRIBUTE       RtBrick-Subscriber-Ifl              26  string

ATTRIBUTE       RtBrick-Terminate-Code              27  integer

ATTRIBUTE       RtBrick-DHCPv4-Client-Options       30  octets
ATTRIBUTE       RtBrick-DHCPv4-Server-Options       31  octets
ATTRIBUTE       RtBrick-DHCPv6-Client-Options       32  octets
ATTRIBUTE       RtBrick-DHCPv6-Server-Options       33  octets

ATTRIBUTE       RtBrick-L2TP-Pool                   40  string
ATTRIBUTE       RtBrick-L2TP-Profile                41  string
ATTRIBUTE       RtBrick-L2TP-Tx-Connect-Speed       42  integer
ATTRIBUTE       RtBrick-L2TP-Rx-Connect-Speed       43  integer

ATTRIBUTE       RtBrick-QoS-Profile                 62  string
ATTRIBUTE       RtBrick-QoS-Shaper                  63  string
ATTRIBUTE       RtBrick-QoS-Parent-Scheduler        64  string
ATTRIBUTE       RtBrick-QoS-Policer                 65  string
ATTRIBUTE       RtBrick-QoS-MFC                     66  string
ATTRIBUTE       RtBrick-QoS-Queues                  67  string

ATTRIBUTE       RtBrick-AAA-Profile                 69  string
ATTRIBUTE       RtBrick-Service-Profile             70  string

ATTRIBUTE       RtBrick-IGMP-Status                 71  integer
ATTRIBUTE       RtBrick-IGMP-Profile                72  string
ATTRIBUTE       RtBrick-IGMP-Version                73  integer
ATTRIBUTE       RtBrick-IGMP-Max-Members            74  integer

ATTRIBUTE       RtBrick-HTTP-Redirect-URL           75  string
ATTRIBUTE       RtBrick-IPv4-ACL-In                 76  string
ATTRIBUTE       RtBrick-IPv4-ACL-Out                77  string
ATTRIBUTE       RtBrick-IPv6-ACL-In                 78  string
ATTRIBUTE       RtBrick-IPv6-ACL-Out                79  string

ATTRIBUTE       RtBrick-Class-0-Packets-Out         81  integer64
ATTRIBUTE       RtBrick-Class-0-Bytes-Out           82  integer64
ATTRIBUTE       RtBrick-Class-1-Packets-Out         83  integer64
ATTRIBUTE       RtBrick-Class-1-Bytes-Out           84  integer64
ATTRIBUTE       RtBrick-Class-2-Packets-Out         85  integer64
ATTRIBUTE       RtBrick-Class-2-Bytes-Out           86  integer64
ATTRIBUTE       RtBrick-Class-3-Packets-Out         87  integer64
ATTRIBUTE       RtBrick-Class-3-Bytes-Out           88  integer64
ATTRIBUTE       RtBrick-Class-4-Packets-Out         89  integer64
ATTRIBUTE       RtBrick-Class-4-Bytes-Out           90  integer64
ATTRIBUTE       RtBrick-Class-5-Packets-Out         91  integer64
ATTRIBUTE       RtBrick-Class-5-Bytes-Out           92  integer64
ATTRIBUTE       RtBrick-Class-6-Packets-Out         93  integer64
ATTRIBUTE       RtBrick-Class-6-Bytes-Out           94  integer64
ATTRIBUTE       RtBrick-Class-7-Packets-Out         95  integer64
ATTRIBUTE       RtBrick-Class-7-Bytes-Out           96  integer64

ATTRIBUTE       RtBrick-Policer-L1-Packets-In       97  integer64
ATTRIBUTE       RtBrick-Policer-L1-Bytes-In         98  integer64
ATTRIBUTE       RtBrick-Policer-L2-Packets-In       99  integer64
ATTRIBUTE       RtBrick-Policer-L2-Bytes-In        100  integer64
ATTRIBUTE       RtBrick-Policer-L3-Packets-In      101  integer64
ATTRIBUTE       RtBrick-Policer-L3-Bytes-In        102  integer64
ATTRIBUTE       RtBrick-Policer-L4-Packets-In      103  integer64
ATTRIBUTE       RtBrick-Policer-L4-Bytes-In        104  integer64

ATTRIBUTE       RtBrick-DNS-Primary                131  ipaddr
ATTRIBUTE       RtBrick-DNS-Secondary              132  ipaddr
ATTRIBUTE       RtBrick-DNS-Primary-IPv6           134  ipv6addr
ATTRIBUTE       RtBrick-DNS-Secondary-IPv6         135  ipv6addr

ATTRIBUTE       RtBrick-Instance                   137  string
ATTRIBUTE       RtBrick-Gateway-Ifl                138  string

ATTRIBUTE       RtBrick-Connection-Status-Message   139  string

ATTRIBUTE       RtBrick-LI-Action                  140  integer encrypt=2
ATTRIBUTE       RtBrick-LI-Identifier              141  integer encrypt=2
ATTRIBUTE       RtBrick-LI-Direction               142  integer encrypt=2
ATTRIBUTE       RtBrick-LI-MED-Instance            143  string  encrypt=2
ATTRIBUTE       RtBrick-LI-MED-IP                  144  ipaddr  encrypt=2
ATTRIBUTE       RtBrick-LI-MED-Port                145  integer encrypt=2

#         VALUE MAPS
#
#       Attribute               Name        Number
VALUE   RtBrick-LI-Action       NOOP        0
VALUE   RtBrick-LI-Action       ON          1
VALUE   RtBrick-LI-Action       OFF         2

VALUE   RtBrick-LI-Direction    INGRESS     1
VALUE   RtBrick-LI-Direction    EGRESS      2
VALUE   RtBrick-LI-Direction    BOTH        3

VALUE   RtBrick-IGMP-Status     DISABLED    0
VALUE   RtBrick-IGMP-Status     ENABLED     1

VALUE   RtBrick-IGMP-Version    V1          1
VALUE   RtBrick-IGMP-Version    V2          2
VALUE   RtBrick-IGMP-Version    V3          3

END-VENDOR RtBrick
```

- The **RtBrick-DHCP-Client-Options** and **RtBrick-DHCPv6-Client-Options** attributes enable the transmission of DHCPv4/DHCPv6 options in the Access-Accept message as payload. These options are added to packets sent to the client, operating in relay/proxy mode or server mode.

- The **RtBrick-DHCP-Server-Options** and **RtBrick-DHCPv6-Server-Options** attributes are only supported in relay/proxy mode and are used to add options in packets sent toward the DHCP server.

5.2.10. Appendix B - RADIUS Standard Attributes Supported

Disclaimer Note: The following table presents the current development state of the RADIUS attributes supported in RBFS. Some of these may be incomplete and are subjected to change at any time without any notice.

RFC	**Attribute**		**Access**		**Accounting**				**CoA**	
	#	Name	Request	Response	Acct-Start	Acct-Interim	Acct-Stop	Acct-On	Request	Response

2865	

	Messa ge								
25	Class		X	X	X	X			
32	NAS-Identifier	X		X	X	X	X		
61	NAS-Port-Type	X		X	X	X			

2688	40	Acct-Status-Type			X	X	X	X		
	41	Acct-Delay-Time								
	42	Acct-Input-Octets				X	X			
	43	Acct-Output-Octets				X	X			
	44	Acct-Session-ID	X		X	X	X			
	45	Acct-Authentic								
	46	Acct-Session-Time					X			
	47	Acct-Input-Packets				X	X			
	48	Acct-Output-Packets				X	X			
	49	Acct-Terminate-Cause					X			

2869	52	Acct-Input-Gigawords				X	X			
	53	Acct-Output-Gigawords				X	X			
	55	Event-Timestamp			X	X	X			
2865	60	CHAP-Challenge	X							

2868	64	Tunnel-Type		X	X	X	X			
	65	Tunnel-Medium-Type		X						
	66	Tunnel-Client-Endpoint		X	X	X	X			
	67	Tunnel-Server-Endpoint		X	X	X	X			
	69	Tunnel-Password		X						
	81	Tunnel-Private-Group-ID		X	X	X	X			
	82	Tunnel-Assignment-ID		X	X	X	X			
	83	Tunnel-Preference		X					X	

2869	85	Acct-Interim-Interval	X						X	
	87	NAS-Port-Id	X		X	X	X			
	88	Framed-Pool		X						
2868	90	Tunnel-Client-Auth-ID		X	X	X	X			
	91	Tunnel-Server-Auth-ID		X	X	X	X			
3162	97	Framed-IPv6-Prefix		X	X	X	X			
	100	Framed-IPv6-Pool		X						
3576	101	Error-Cause								X
4818	123	Delegated-IPv6-Prefix		X	X	X	X			

6911	171	Delegated-IPv6-Prefix-Pool		X						

4679	

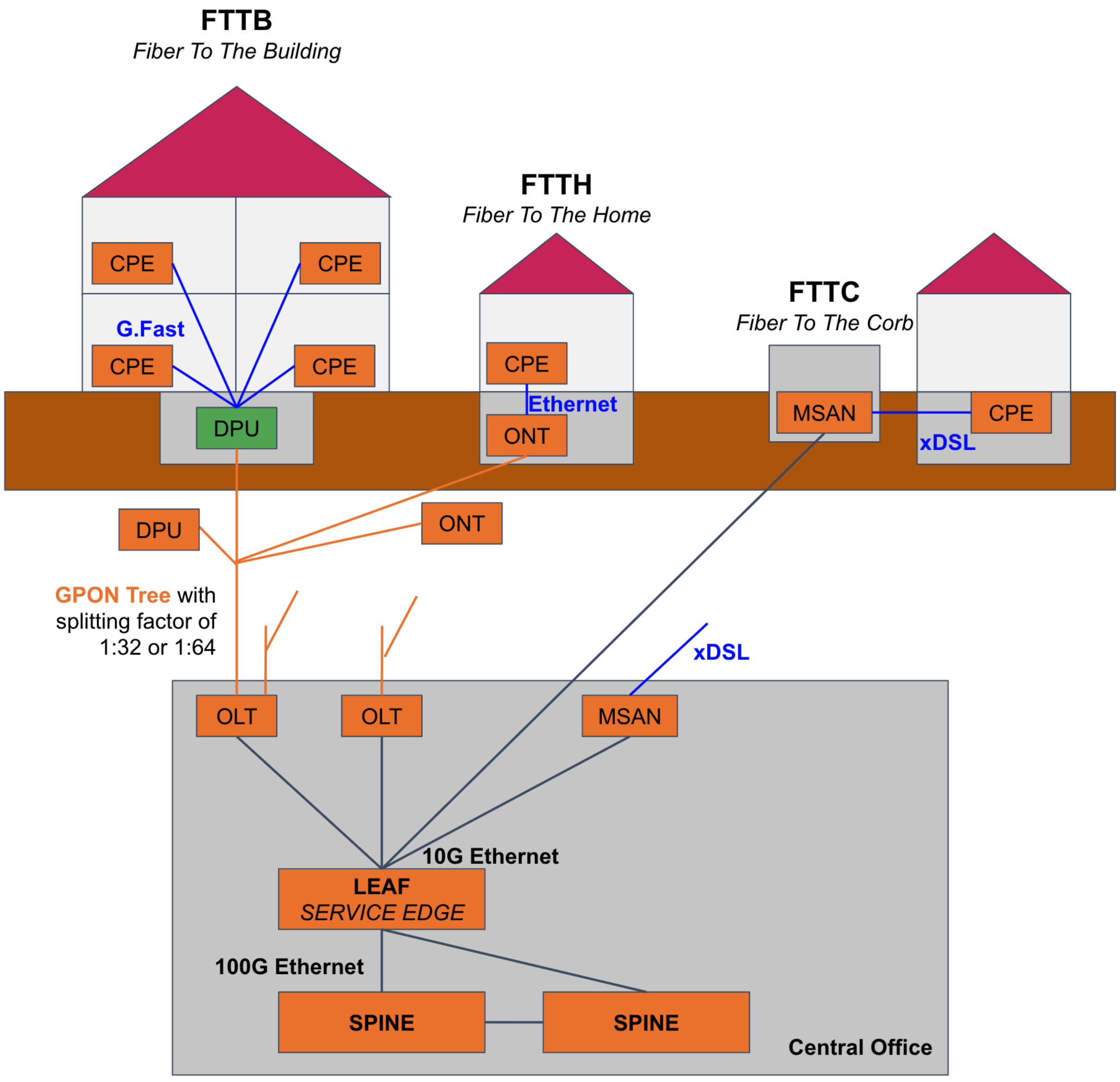
	m-Low-Power								
26-3561-139	Maximum-Interleaving-Delay-Upstream	X	X	X	X	X			
26-3561-140	Actual-Interleaving-Delay-Upstream	X	X	X	X	X			
26-3561-141	Maximum-Interleaving-Delay-Downstream	X	X	X	X	X			
26-3561-142	Actual-Interleaving-Delay-Downstream	X	X	X	X	X			
26-3561-144	Access-Loop-Encapsulation	X	X	X	X	X			

RtBrick	

	26-50058-140	RtBrick-LI-Action		X					X	
	26-50058-141	RtBrick-LI-Identifier		X					X	
26-50058-142	RtBrick-LI-Direction		X					X		26-50058-143
RtBrick-LI-MED-Instance		X					X		26-50058-144	RtBrick-LI-MED-IP
	X					X		26-50058-145	RtBrick-LI-MED-Port	

5.2.11. Appendix C - Access Deployments

The following picture shows different deployments typically used in access networks.



The LEAF and SPINE are bare metal switches running RBFS where the LEAF is responsible for service edge functionalities like PPPoE subscriber termination and the SPINE builds the actual fabric core. Multiple LEAF switches will be connected to a pair of spine switches.

5.3. RBFS Carrier-Grade Network Address Translation

5.3.1. Carrier-Grade Network Address Translation Overview

Carrier-Grade NAT

The rise of technologies and the rapid growth of mobile devices and cloud services worldwide resulted in the exhaustion of IPv4 addresses. Broadband service providers face a growing challenge as this 32-bit address is insufficient to meet the ever-growing demand and the cost of obtaining public IP addresses for every individual subscriber. Though the launch of IPv6 can address the IPv4 depletion problem, migrating to an IPv6 network becomes vastly complex and costly. IPv6 also does not provide backward compatibility with IPv4. Consequently, migrating to IPv6 is not an ideal solution for many broadband service providers.

Carrier-Grade Network Address Translation (CGNAT) is a prominent technology solution that addresses the IPv4 exhaustion challenge of service providers. The solution helps to use private network addresses and to limit the use of publicly routable IPv4 addresses significantly.

CGNAT allows service providers to serve a large number of their subscribers using a limited number of public IPv4 addresses. It saves costs on public IPv4 addresses and conserves their IPv4 address pools.

Static NAT

Static NAT associates one private IP address with one public IP address. This is a one-to-one mapping and you must manually define the mapping between a private and public address.

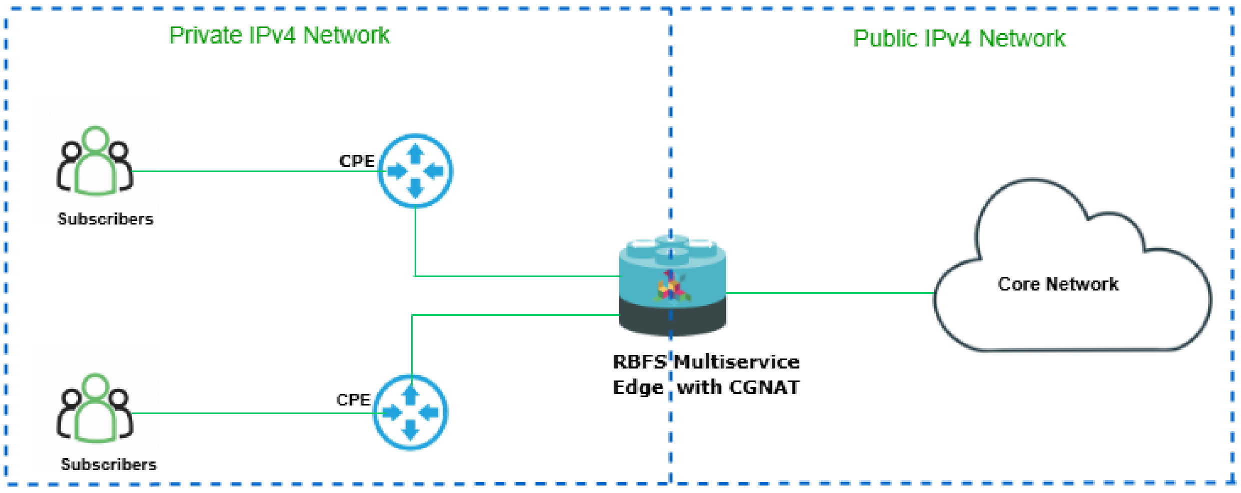
Dynamic NAT

In Dynamic NAT, internal private IPv4 addresses are mapped with a public IPv4 address and this mapping of a private IPv4 address to a public IPv4 address happens dynamically. The router dynamically picks an address from the address pool that is not currently assigned.

NAT444

CGNAT supports NAT444. NAT 444 refers to three sets of IPv4 addresses: Customer private, ISP private, and public Internet. Network address translation occurs from customer private to ISP private, and then ISP private to public. The NAT444, also known as CGNAT, functionality maps IPv4 subscriber private addresses to IPv4 public addresses.

The following diagram illustrates a high-level view of CGNAT.



CGNAT Benefits

CGNAT allows service providers to provide services to a large number of their subscribers using a limited number of public IPv4 addresses. The benefits include:

Conserve Public IPv4 Addresses

The solution helps service providers conserve expensive public IPv4 addresses by enabling multiple subscribers to share a single public IPv4 address. With CGNAT, one public IPv4 address can manage hundreds of devices within the private network. It reduces the requirement of continually buying additional public IP addresses.

Enhance Security

In addition to stretching the limited pool of public IPv4 addresses even further, CGNAT also provides significant security benefits. It helps to enhance network security by keeping the internal addressing private from the external network. It makes it difficult for attackers to target specific devices on the network by preventing attacks that target specific IP addresses.

Performance at Carrier-Grade

The CGNAT solution implements the functions of IPv4 address translation in the network of service providers. Service providers can deploy NAT in a way that allows multiple subscribers to share a single global IPv4 address and scale to several thousands of address translations. CGNAT operations are implemented at the forwarding plane (hardware level) to ensure optimal performance. This hardware-level implementation achieves high throughput with no performance impact and packet processing occurs without overburdening the CPU. The solution offers carrier-grade scalability with fast translation rates and a large number of IPv4 address and port number translations.

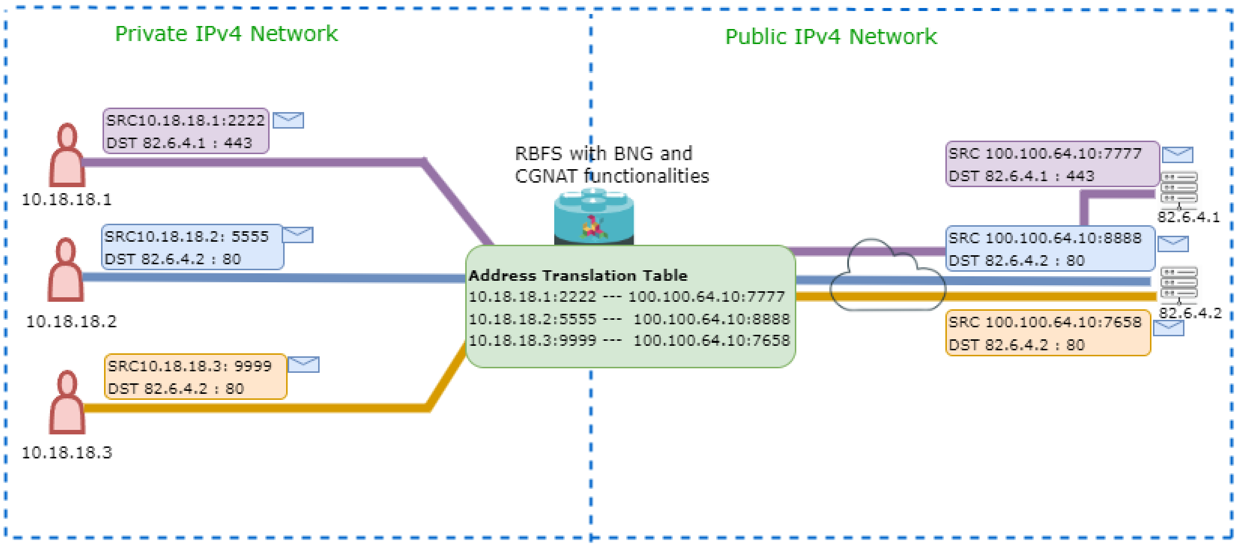
Understanding CGNAT Implementation

CGNAT solution has been designed to support Port Address Translation, also known as Network Address Port Translation (NAPT), which has the greatest potential to conserve IPv4 addresses for service providers. NAPT is known as an effective method to allow multiple devices to connect to the Internet using a single public IPv4 address.

Network Address Port Translation

Network Address Port Translation is a dynamic NAT in which port numbers along with IP address are used to identify which traffic belongs to which private IP address. CGNAT translates the source private IPv4 address and port number to a public source IPv4 address and unique port number. This allows multiple devices with different private IP addresses to use a single public IPv4 address. The unique port number ensures that the traffic is delivered to the correct device.

The following diagrams illustrate how CGNAT works at a high level.



The diagram shows three subscribers using private IPv4 addresses (10.18.18.1, 10.18.18.2, and 10.18.18.3 send traffic to two different servers (82.6.4.1 and 82.6.4.2) on the public network.

BNG with CGNAT performs the address translation by replacing the source private IPv4 address with the public IPv4 address from the address pool and the source port number with a unique port number. After the address translation, the device forwards the traffic to the destination servers. After the translation, packets have a new source IPv4 address which is the public IPv4 and a unique port number as per the mapping in the address translation table.

CGNAT maintains a translation table with the entries. Entries are records of mapped private IPv4 addresses and port numbers with the public IPv4 addresses and port numbers.

The downstream traffic from the servers traverses to the CGNAT device. The packets coming from external hosts include the destination address as the (translated) public IPv4 address. CGNAT device performs reversal address translation for these packets as per the translation table mapping for the downstream traffic.

NAPT helps to significantly reduce the number of logs generated as it generates logs only during the allocation and release of each block of ports.

Deterministic NAT

CGNAT offers support for deterministic NAT mode, which provides a consistent mapping of private IPv4 addresses with public IPv4 addresses and port ranges. This mode ensures a one-to-one mapping of private IPv4 addresses with public IPv4 addresses, allowing you to specify the private address and its matching public address and port range. The given private IPv4 address is always translated to the same public address.

In addition, deterministic NAT guarantees a predetermined and fixed translation for a given internal address with a public IPv4 and port combination, ensuring consistency and stability in the mapping. This feature is particularly useful for applications that involve security protocols, as well as for service providers who need to track subscriber sessions.

Furthermore, deterministic NAT significantly reduces address translation logs, as private IPv4 addresses are always mapped to public IPv4 addresses and port ranges.

Address Translation Table

Port mapping is a feature that allows multiple devices to share the same public IPv4 address with different port numbers. CGNAT generates a unique port number for each subscriber session. Port mapping helps to determine the correct host among the many devices in the private network that use the same public IPv4 address.

CGNAT maintains an address translation table that maps private IPv4 addresses and port numbers to their corresponding public IPv4 addresses and ports. Whenever an incoming packet arrives, CGNAT checks the translation table to see if a translation entry exists for that packet. If there's an entry, the CGNAT replaces the source IPv4 address and port number in the packet header with the mapped public IPv4 address and port number.

When a device on the Internet sends packets downstream, the CGNAT software uses the translation table for address translation reversal. It replaces the destination public IPv4 address and port number in the packet header with the corresponding private IPv4 address and port number.

NAT IP Pools and Chaining

A NAT pool contains a range of multiple public IP addresses. You can create multiple pools and associate them with a NAT service. Pool chaining is a method in which you can associate one pool with another. For a pool, you can define 'next pool name'; so that when the pool gets exhausted with the IPv4 addresses, the next pool that is defined will take over.

Aging

Aging refers to the time that a translation entry exists or remains in the address translation table after it was last used.

The entries in the translation table have a finite lifespan as the software implements mechanisms to handle session timeouts.

When a host sends a packet to a destination, CGNAT translates the private IPv4 address and port to a public IPv4 address and port. This mapping is recorded as an entry in the address translation table. The software always looks up the translation table whenever it receives a packet to verify that any entry exists for the packet in the translation table. The software performs the address translation based on the recorded entries and its associated mappings for both inbound and outbound packets.

If there is no activity related to a specific mapping (for example, when there are no incoming or outgoing packets), the mapping will eventually be removed from the address translation table, once idle or unused entries are detected, entries removed from the address translation table to free up resources for the new upcoming traffic flows.

For TCP, idle or unused flows are typically detected by the receipt of TCP FIN-ACK packets, which indicate that both sides of the connection have finished sending data. Once detected, these flows are removed after a configurable aging timer expires.

For UDP, as it is a connectionless protocol, idle or unused flows are detected by periodically polling the flow traffic. As the number of UDP traffic flows increases, the idle flow detection mechanism also increases, ensuring that idle flows are removed in a timely manner. The aging of UDP flows is proportional to the number of traffic flows.

Ageing Timeout for UDP-based DNS Traffic

You can define an ageing timeout that controls how long the system maintains state information for UDP-based DNS traffic before considering the flow inactive and clearing it from the NAT table.

UDP traffic has a default ageing timeout of 30 seconds in NAT implementation.

```
NOTE: The NAT ageing timeout is 200 seconds for TCP and UDP traffic. The actual
time taken to flush aged-out NAT rules depends on the number of NAT rules and the
number of counters.
```

Support for IPoE N:1 Shared VLAN in Single VRF

CGNAT is supported for the IPoE N:1 shared VLAN model within the same VRF instance.

Logging

CGNAT provides traffic monitoring capabilities to track and log network activities. The solution offers a flexible logging mechanism that can store information such as port numbers, time, destination, and address translation details. You can enable logging for CGNAT operations.

NAT for ICMP Messages

The NAT-enabled RBFS devices can perform network address translation for ICMP echo request and reply messages. Subscribers can use ping command (ICMP Echo Request, Echo Reply) to check internet connectivity to external internet addresses when NAT is enabled. This feature is enabled by default.

NAT for TCP Control Messages

RBFS with NAT supports address translation for TCP control messages using ACLs. It includes the TCP control messages such as FIN, FIN-ACK, RST, RST-ACK, SYN, and SYN-ACK. This feature is enabled by default.

NAT-miss Traffic Handling in Control Plane on Q2C Platforms

RBFS supports a mechanism to handle NAT-miss packets at control plane. A NAT-miss occurs when a downstream packet arrives at a device configured for address translation, but no matching NAT rule exists for that flow. A new ACL rule issued to detect such packets based on NAT-miss status and destination prefix match. These packets are redirected to the control plane, which performs network address translation in software and forwards them.

Guidelines and Limitations

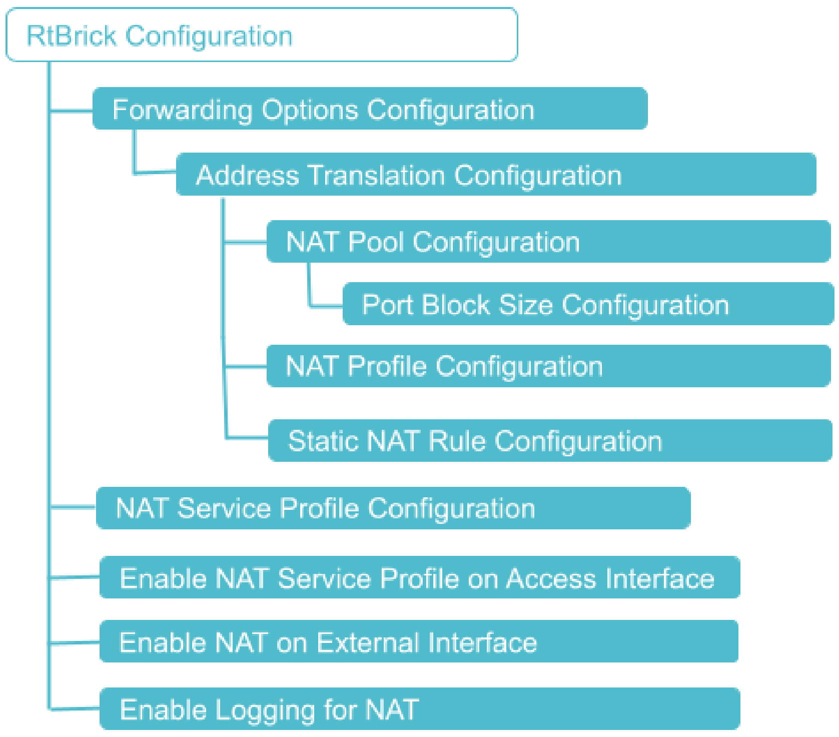
- On Q2A platforms, avoid installing IP forwarding routes that overlap with the public IP address range used for CGNAT.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

5.3.2. Carrier-Grade NAT Configuration

Configuration Hierarchy


RtBrick Configuration
Forwarding Options Configuration
Address Translation Configuration
NAT Pool Configuration
Port Block Size Configuration
NAT Profile Configuration
Static NAT Rule Configuration
NAT Service Profile Configuration
Enable NAT Service Profile on Access Interface
Enable NAT on External Interface
Enable Logging for NAT

CGNAT Configuration

You must perform the following tasks to configure CG NAT.

1. Configure NAT Pool
2. Configure NAT Profile
3. Configure NAT Rule
4. NAT Service Profile Configuration
5. Enable NAT on an Instance
6. Enable NAT Service Profile on Access Interface
7. Enable NAT on External Interface
8. Enable Logging for NAT
9. Enabling NAT ACL Support for ICMP and TCP Control Messages

Configuration Syntax and Commands

The following sections describe the CGNAT configuration syntax and commands.

Configuring NAT Profile

A NAT profile defines how the NAT device has to perform the IPv4 address translation. NAT profile allows you to define an instance, IPv4 address pools, maximum number of translations, and mapping a particular internal IPv4 address with a particular external IPv4 address for a deterministic address translation.

You can create NAT profile for an RBFS instance using the 'instance' option. Also, you can define the TCP or UDP traffic type for the profile.

A single NAT profile can be attached to subscribers across different instances. This means that regardless of which instance a subscriber belongs to, they can share the same NAT profile. For example, subscribers in different routing instances can be managed using the same NAT settings.

Subscribers within the same instance can be assigned different NAT profiles. Different groups of subscribers or services within the same instance can have tailored NAT configurations.

Syntax:

set forwarding-options address-translation profile <profile-name> <attribute> <value>

Attribute	Description
<profile-name>	Specify the NAT profile name.
deterministic [true]	Specify deterministic as true to enable deterministic NAT for the profile. Deterministic NAT allows subscribers always to connect with a single public IP.
instance	Specify the RBFS instance.
ip-protocol	Specify the protocol: tcp or udp. Note: When using the tcp-control match attribute in a match rule, it is necessary to explicitly configure match ip-protocol for TCP. This ensures that the rule accurately identifies and processes TCP packets before applying any matching for TCP control flags.
ip-protocol ageing-timeout <ageing-timeout>	Specify the ageing time value for the protocol. Default, 120 seconds. Allowed range: 60 - 86400 seconds.

Attribute	Description
ip-protocol control-plane-ageing-timeout <ageing-timeout>	Specify the TCP control ageing timeout value. Default, 10 seconds. Allowed range: 5 - 120 seconds.
max-rules	Specify the maximum number of rules for address translations for a public IPv4 address and for an interface. Note: The supported values for maximum address translation for an interface are 64, 128, 256, 512, 1024, 2048, 4096, 8192, and 16342.
pool	Specify the name of the public IP address pool.

The following commands configure the NAT profile named nat_profile1. The nat profile nat_profile1 is configured on the instanced vrf1 with a pool attached nataddr_pool1. Maximum rules are configured as 100 rules and the aging period is configured as 600 seconds for TCP traffic and 300 seconds for UDP traffic.

```
set forwarding-options address-translation profile nat_profile1
set forwarding-options address-translation profile nat_profile1 pool nataddr_pool1
set forwarding-options address-translation profile nat_profile1 max-rules 128
set forwarding-options address-translation profile nat_profile1 ip-protocol tcp
ageing-timeout 600
set forwarding-options address-translation profile nat_profile1 ip-protocol udp
ageing-timeout 300
set forwarding-options address-translation profile nat_profile1 ip-protocol other
ageing-timeout 300
```

Example Configuration:

```
supervisor@rtbrick>cbng1.rtbrick.net: cfg> show config forwarding-options address-
translation profile
{
  "rtbrick-config:profile": [
    {
      "profile": "nat_profile1",
      "pool": "nataddr_pool1",
      "max-rules": "120",
      "ip-protocol": {
        "tcp": {
          "ageing-timeout": 600
        },
        "udp": {
          "ageing-timeout": 300
        },
        "other": {
          "ageing-timeout": 300
        }
      }
    }
    ]
 }
```

NAT Pool Configuration

A NAT IP address pool includes a set of public IPv4 addresses that are used for network address translation. You can create multiple public IPv4 address pools and one pool includes a range of public IPv4 addresses. These pools allocate public IPv4 addresses to subscribers during address translation. While configuring a pool, you can define the group of public IPv4 addresses belonging to that pool by specifying the lowest and highest IP addresses.

The system allows you to create multiple pools and define the association among them. You can define the 'next-pool-name' that takes over when the current pool gets exhausted with the IPv4 addresses. When one pool gets exhausted, the next pool takes over and starts serving the IP addresses to subscribers when the address translation occurs.

Syntax:

set forwarding-options address-translation pool <pool-name> <attribute> <value>

Attribute	Description
<pool-name>	Specify the name of the address pool.
ipv4-address	Specify both the highest and lowest IPv4 addresses in the range of IPv4 addresses for the pool.
ipv4-address high	Specify the highest IPv4 address in the address pool. You must specify the highest IP address in the range of IP addresses.
ipv4-address low	Specify the lowest IPv4 address in the address pool. You must specify the lowest IP address in the range of IP addresses.
next-pool-name	Specify the name of the next address pool that is to be used when the current address pool is allocated completely.

Example Configuration:

The following commands configure nataddr_pool1 as the NAT pool and nataddr_pool2 as the next pool.

It indicates NAT pool nataddr_pool1 contains a range of public IPv4 addresses from 100.100.100.1 to 100.100.100.5.

When the pool nataddr_pool1 has fully allocated its IPv4 addresses, the next pool named nataddr_pool2 will start allocating IPv4 addresses from its pool. The pool nataddr_pool2 includes a rage of IPv4 address from 100.100.101.1 to 100.100.101.150.

```
set forwarding-options address-translation pool nataddr_pool1
set forwarding-options address-translation pool nataddr_pool1 next-pool-name
nataddr_pool2
set forwarding-options address-translation pool nataddr_pool1 ipv4-address low
100.100.100.1
set forwarding-options address-translation pool nataddr_pool1 ipv4-address high
100.100.100.5
set forwarding-options address-translation pool nataddr_pool2 ipv4-address low
100.100.101.1
set forwarding-options address-translation pool nataddr_pool2 ipv4-address high
100.100.101.150
```

Example Configuration:

```
supervisor@rtbrick: cfg> show config forwarding-options address-translation pool
{
  "rtbrick-config:pool": [
    {
      "pool-name": "nataddr_pool1",
      "next-pool-name": "nataddr_pool2",
      "ipv4-address": {
        "low": "100.100.100.1",
        "high": "100.100.100.5"
      }
    },
    {
      "pool-name": "nataddr_pool2",
      "ipv4-address": {
        "low": "100.100.101.1",
        "high": "100.100.101.150"
      }
    }
  ]
}
```

NAT Rule Configuration

You can define NAT rules only for static NAT. A NAT rule defines a match condition and a corresponding action. After you specify NAT rules, each packet is matched with each NAT rule. If a packet matches the condition specified in a rule, then the action corresponding to that match occurs. Match rules govern how the translation of private IPv4 addresses to public IPv4 addresses is performed.

With NAT rules, you can define how address translation is applied to traffic, and how to handle various protocols and data traffic, such as TCP and UDP, to ensure proper address translation and the mappings of private addresses to public addresses.

Rules also define how to handle inbound and outbound traffic, different protocols, and data traffic such as TCP and UDP for ensuring the proper address translation of traffic.

A maximum of 16,384 NAT rules on NAT profiles is allowed.

Syntax:

set forwarding-options address-translation rule <rule-name> <attribute> <value>

Attribute	Description
<rule-name>	Specify the name of the rule.
ordinal <ordinal-value>	Specify the ordinal value. An ordinal value is a numerical representation that indicates its relative position or order.
ordinal <ordinal-value> instance	Specify the RBFS instance name.
ordinal <ordinal-value> ip-protocol [tcp/udp]	Specify the IP protocol, TCP or UDP.
ordinal <ordinal-value> local [ipv4-address/port]	Specify the private IPv4 address or port number that needs to be translated.
ordinal <ordinal-value> public [ipv4-address/port]	Specify the public IPv4 address. This public IP will be mapped with the private IP in the translation table.

Configuring NAT Service Profile

You must create a NAT service profile and attach the service profile with the access interface for enabling CGNAT on the interface.

Syntax:

set access service-profile <profile-name> <attribute> <value>

Attribute	Description
<profile-name>	Name of the service profile.
profile <profile>	Specify the profile name for the address translation.

Enable NAT Service Profile on the Access Interface

It is required to attach the NAT service profile to the access interface for enabling address translation on the interface.

Syntax:

set access interface [double-tagged | single-tagged | untagged] <interface-name> <outer-vlan-min> <outer-vlan-max> <inner-vlan-min> <inner-vlan-max> **service-profile-name** <service-profile-name>

Attribute	Description
service-profile	Configure global service profile.
<outer-vlan-min>	Specify the minimum number of outer VLANs. Allowed range: 1 - 4094.
<outer-vlan-max>	Specify the maximum number of outer VLANs. Allowed range 1 - 4094.
<inner-vlan-min>	Specify the minimum number of inner VLANs. Allowed range: 1 - 4094.
<inner-vlan-max>	Specify the maximum number of inner VLANs. Allowed range 1 - 4094.
service-profile-name	Specify the name of the service profile.

The following commands attach service profile nat_service to the interface (double-tagged) ifp-0/0/17. The configuration shows the minimum number of outer VLANs as 1000, the maximum number of outer VLANs as 1007, the minimum number of inner VLANs as 84 and the maximum number of inner VLANs as 4084. The access type configured is IPoE and the service profile is NAT service. AAA profile name is ipoe-aaa and the gateway IFL is lo-0/0/0/100.

IPoE subscriber with outer VLAN between 1000 and 1007, and inner VLAN between 84 and 4084 will be matched with this NAT service profile and a corresponding action will be taken. Anything outside these vlans will not have any action from this NAT service profile.

```
set access interface double-tagged ifp-0/0/17 1000 1007 84 4084
set access interface double-tagged ifp-0/0/17 1000 1007 84 4084 access-type IPoE
set access interface double-tagged ifp-0/0/17 1000 1007 84 4084 access-profile-
name ipoe
set access interface double-tagged ifp-0/0/17 1000 1007 84 4084 service-profile-
name nat_service
set access interface double-tagged ifp-0/0/17 1000 1007 84 4084 aaa-profile-name
ipoe-aaa
set access interface double-tagged ifp-0/0/17 1000 1007 84 4084 gateway-ifl lo-
0/0/0/100
```

```
{
  "rtbrick-config:interface": {
    "double-tagged": [
      {
        "interface-name": "ifp-0/0/17",
        "outer-vlan-min": 1000,
        "outer-vlan-max": 1007,
        "inner-vlan-min": 84,
        "inner-vlan-max": 4084,
        "access-type": "IPoE",
        "access-profile-name": "ipoe",
        "service-profile-name": "nat_service",
        "aaa-profile-name": "ipoe-aaa",
        "gateway-ifl": "lo-0/0/0/100"
      }
    ]
  }
}
```

The following commands configure a double-tagged interface ifp-0/0/6 and outer VLAN minimum value is 1000, maximum value as 1007, inner VLAN minimum value as 84 and maximum value as 4084. The access type configured is PPPoE, service profile is NAT service. AAA profile name is configured as ipoe-aaa.

It indicates PPPoE subscriber with outer VLAN between 1000 and 1007, and inner VLAN between 84 and 4084 will be matched with this NAT service profile and a corresponding action will be taken. Anything outside these vlans will not have any action from this NAT service profile.

```
set access interface double-tagged ifp-0/0/16 1000 1007 84 4084
set access interface double-tagged ifp-0/0/16 1000 1007 84 4084 access-type PPPoE
set access interface double-tagged ifp-0/0/16 1000 1007 84 4084 access-profile-
name pppoe
set access interface double-tagged ifp-0/0/16 1000 1007 84 4084 service-profile-
```

```
name nat_service
set access interface double-tagged ifp-0/0/16 1000 1007 84 4084 aaa-profile-name
ipoe-aaa
```

Example:

```
{
  "rtbrick-config:interface": {
    "double-tagged": [
      {
        "interface-name": "ifp-0/0/16",
        "outer-vlan-min": 1000,
        "outer-vlan-max": 1007,
        "inner-vlan-min": 84,
        "inner-vlan-max": 4084,
        "access-type": "PPPoE",
        "access-profile-name": "pppoe",
        "service-profile-name": "nat_service",
        "aaa-profile-name": "ipoe-aaa"
      },
    ]
  }
}
```

Enable NAT on an Instance

You can enable address translation for a specific routing instance such as VRF or virtual router rather than globally across the entire router.

Syntax:

set instance <instance-name> **address-translation true**

Attribute	Description
<instance-name>	Name of the instance.

The following command is used to enable address translation for the instance named 'vrf1'.

```
set instance vrf1 address-translation true
```

```
supervisor@rtbrick.net: op> show config instance vrf1 address-translation
{
  "rtbrick-config:address-translation": "true"
}
```

Enable NAT on External Interface

It is required to enable the NAT on the external (core-facing) interface.

Syntax:

set interface <interface-name> **unit** <unit-id> **address-translation direction** <public>

Attribute	Description
<interface-name>	Name of the interface.
<unit-id>	Configure the number of sub-interfaces under the physical interface.
public	Specify 'public' for the external interface.

The following commands configure the external interface ifp-0/1/64 for IPv4 address translation. 'Unit' logical identifier for this physical interface. Direction 'public' shows the configuration on external interface for address translation.

```
set interface ifp-0/1/64 unit 100
set interface ifp-0/1/64 unit 100 address-translation
set interface ifp-0/1/64 unit 100 address-translation direction public
```

```
supervisor@rtbrick.net: cfg> show config interface ifp-0/1/64 unit 100
{
  "rtbrick-config:unit": [
    {
      "unit-id": 100,
      "address-translation": {
        "direction": "public"
      }
    }
  ]
}
```

Enable Logging for NAT

You can optionally enable logging for CGNAT operations.

All RBFS logs and related information is available in the *RBFS Logging User Guide*. For the list of RBFS logs, see Log Reference.

set log bd <name> <options>

Attribute	Description
level	Specify the log level.
module	Specify the log module.
plugin-alias	Specify the plugin-alias URL. Plugin-alias is an external logging host server to which you can export logs. For example, Graylog.

The following commands configure logging for NAT module natd with a log level 'debug'.

```
set log bd natd
set log bd natd module nat
set log bd natd module nat level debug
```

```
supervisor@rtbrick.net: cfg> show config log bd natd
{
  "rtbrick-config:bd": [
    {
      "bd-name": "natd",
      "module": [
        {
          "module-name": "nat",
          "level": "debug"
        }
      ]
    }
  ]
}
```

Enabling NAT ACL Support for ICMP and TCP Control Messages

You can enable Access Control Lists support for ICMP and TCP Control messages.

Enabling NAT Support for ICMP Using ACLs

The NAT ACLs for ICMP packets allows address translation for ICMP request and response packets through the use of Access Control Lists (ACLs). This enhances the handling of ICMP traffic by providing precise control over NAT behavior for ICMP packets.

Setting up the NAT ACLs for ICMP will be handled by the NAT application with the 25.1.1 release, eliminating the need for manual configuration.

set forwarding-options acl l3v4 rule rule-name> ordinal <ordinal-number> <attribute > <value>

Attribute	Description
<rule-name>	Specify the name of the rule.
ordinal <ordinal-value>	Specify the ordinal value. An ordinal value is a numerical representation that indicates its relative position or order.
match	Define the conditions under which the rule will be applied. These conditions determine which packets the rule should act on.
action	Defines what happens to packets that match the rule’s criteria.
priority	Determines the priority of the rule when multiple rules with the same ordinal value are evaluated.
direction	The rule applies to ingress or egress traffic.

Example commands:

```
set forwarding-options acl l3v4 rule NAT_TRAP
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 match ip-protocol icmp
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 match destination-ipv4-
prefix-list PREFIXES_NAT_PUBLIC
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 priority 1001

set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 match source-ipv4-
prefix 100.64.0.0/10
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 match ip-protocol icmp
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 priority 1002
```

Example:

```
supervisor@rtbrick>rtbrick.net: cfg> show config forwarding-options acl l3v4
{
  "rtbrick-config:l3v4": {
    "rule": [
      {
        "rule-name": "NAT_TRAP",
        "ordinal": [
          {
            "ordinal-value": 1001,
            "match": {
              "direction": "ingress",
              "ip-protocol": "icmp",
              "destination-ipv4-prefix-list": "PREFIXES_NAT_PUBLIC"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1001
          },
          {
            "ordinal-value": 1002,
            "match": {
              "direction": "ingress",
              "source-ipv4-prefix": "100.64.0.0/10",
              "ip-protocol": "icmp"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1002
          }
        ]
      }
    ]
      }
    }
```

The preceding configuration defines Layer 3 IPv4 Access Control List (ACL) rules for managing ICMP traffic. The rule with the ordinal value 1001 is designed to redirect ICMP traffic for Network Address Translation (NAT) processing based on specific criteria. The ACL rule is named NAT_TRAP.

The matching criteria for this rule include the direction of traffic, which is set to ingress. This means the rule applies only to incoming traffic. The specified protocol is ICMP, indicating that the rule is applicable to ICMP packets.

The 'destination-ipv4-prefix-list' is designated as PREFIXES_NAT_PUBLIC, which is a predefined list containing a range of IPv4 addresses. The rule specifically pertains to ICMP packets whose destination IP is included in the PREFIXES_NAT_PUBLIC list. These rules ensure that ICMP packets requiring NAT handling are identified and appropriately redirected to the NAT process. When an ICMP packet that meets the conditions (ingress, ICMP, and a destination in PREFIXES_NAT_PUBLIC) is detected, it is redirected (or punted) to the NAT process for further handling.

Enabling NAT Support for TCP Control Messages Using ACL

This configuration enables address translation on TCP control messages through

ACLs. The feature supports control messages such as FIN, SYN, and RST. It applies functionality to TCP control messages such as fin-ack and rst-ack.

Setting up the NAT ACLs for TCP control messages will be handled by the NAT application with the 25.1.1 release, eliminating the need for manual configuration.

Syntax:

set forwarding-options acl l3v4 rule <rune-name> **ordinal** <ordinal-number> **match tcp-control** [fin-ack | rst-ack]

Attribute	Description
<rule-name>	Specify the name of the rule.
ordinal <ordinal-value>	Specify the ordinal value. An ordinal value is a numerical representation that indicates its relative position or order.
match	Define the conditions under which the rule will be applied. These conditions determine which packets the rule should act on.
action	Defines what happens to packets that match the rule's criteria.
priority	Determines the priority of the rule when multiple rules with the same ordinal value are evaluated.
direction	The rule applies to ingress or egress traffic.
tcp-control <fin-ack>	Matches TCP packets with the FIN-ACK control flag set. Note: When using the tcp-control match attribute in a match rule, it is necessary to explicitly configure match ip-protocol for TCP. This ensures that the rule accurately identifies and processes TCP packets before applying any matching for TCP control flags.
destination-ipv4-prefix-list	Name of the prefix list. The list includes predefined destination IP addresses.

Example commands:

```
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 match tcp-control fin-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 match destination-ipv4-
prefix-list PREFIXES_NAT_PUBLIC
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 priority 1003
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 match source-ipv4-
prefix 100.64.0.0/10
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 match tcp-control fin-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 priority 1004
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 match tcp-control rst-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 match destination-ipv4-
prefix-list PREFIXES_NAT_PUBLIC
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 priority 1005
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 match source-ipv4-
prefix 100.64.0.0/10
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 match tcp-control rst-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 priority 1006
```

Example:

```
supervisor@rtbrick>rtbrick.net: cfg> show config forwarding-options acl l3v4
{
  "rtbrick-config:l3v4": {
    "rule": [
      {
        "rule-name": "NAT_TRAP",
        "ordinal": [
          {
            "ordinal-value": 1003,
            "match": {
              "direction": "ingress",
              "tcp-control": "fin-ack",
              "destination-ipv4-prefix-list": "PREFIXES_NAT_PUBLIC"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1003
          },
          {
            "ordinal-value": 1004,
            "match": {
              "direction": "ingress",
              "source-ipv4-prefix": "100.64.0.0/10",
              "tcp-control": "fin-ack"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1004
          },
          {
            "ordinal-value": 1005,
            "match": {
              "direction": "ingress",
              "tcp-control": "rst-ack",
              "destination-ipv4-prefix-list": "PREFIXES_NAT_PUBLIC"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1005
          },
          {
            "ordinal-value": 1006,
            "match": {
              "direction": "ingress",
              "source-ipv4-prefix": "100.64.0.0/10",
              "tcp-control": "rst-ack"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1006
          }
        ]
      }
    ]
  }
}
```

The preceding configuration defines ACL for Layer 3 IPv4 traffic. The ACL is named NAT_TRAP and consists of multiple rules that match specific traffic patterns and apply a 'trap' action to punt the traffic for further NAT processing. Each rule is identified by an ordinal value.

Match Criteria includes direction which is specified as ingress. All rules are applied to ingress traffic. Protocol is defined as TCP traffic with specific control flags: fin-ack, rst-ack, and rst.

In the configuration:

- Ordinal 1003 matches TCP traffic with the fin-ack flag and a destination IP in the PREFIXES_NAT_PUBLIC list.
- Ordinal 1004 matches TCP traffic with the fin-ack flag and a source IP in the 100.64.0.0/10 range.
- Ordinal 1005 matches TCP traffic with the rst-ack flag and a destination IP in

the PREFIXES_NAT_PUBLIC list.

- Ordinal 1006 matches TCP traffic with the rst-ack flag and a source IP in the 100.64.0.0/10 range.

5.3.3. Carrier Grade NAT Operational Commands

Carrier Grade NAT Show Commands

The CGNAT operational commands provide detailed information about the address translation operations.

NAT Pool Information

This command displays information about IPv4 address pool for a user.

Syntax:

show address-translation allocation <options>

Option	Description
pool	Specify the pool name.
user	Specify the user name.

Example for NAT pool allocation.

```
supervisor@rtbrick.net: cfg> show address-translation allocation pool
Pool: pool1
  Profile: nat_profile1
  Instance: vrf1
  Allocated: 100.00%
Pool: pool2
  Profile: nat_profile1
  Instance: vrf1
  Allocated: 100.00%
supervisor@rtbric
```

Example for allocation details for subscribers

```
supervisor@rtbrick.net: cfg> show address-translation allocation user
User                              Original Address     Translated Address   Port
Range
ppp-0/1/20/72339069014638594      10.100.128.1         100.100.100.100      14912 -
14975
ppp-0/1/20/72339069014638594      10.100.128.1         100.100.100.100      18944 -
```

```
19007
ppp-0/1/20/72339069014638595     10.100.128.3         100.100.100.101      14976 -
15039
ppp-0/1/20/72339069014638595     10.100.128.3         100.100.100.101      18880 -
18943
ppp-0/1/20/72339069014638596     10.100.128.5         100.100.100.101      15104 -
15167
ppp-0/1/20/72339069014638596     10.100.128.5         100.100.100.101      17856 -
17919
ppp-0/1/20/72339069014638597     10.100.128.7         100.100.100.100      15104 -
15167
ppp-0/1/20/72339069014638597     10.100.128.7         100.100.100.100      17792 -
17855
ppp-0/1/20/72339069014638598     10.100.128.9         100.100.100.101      15040 -
15103
ppp-0/1/20/72339069014638598     10.100.128.9         100.100.100.101      17792 -
17855
ppp-0/1/20/72339069014638599     10.100.128.11        100.100.100.100      15040 -
15103
```

NAT Rule Details

Syntax:

show address-translation rule <options>

Option	Description
-	Without any option, the command displays the information for all rules.
instance	Displays NAT rule information for the specified instance.
summary	Displays summary of the information for the NAT rule.
user	Displays user information for the NAT rule.

Example for address translation rule for an instance.

```
supervisor@rtbrick.net: cfg> show address-translation rule instance default
Instance: default
User                              Protocol   Original Address        Translated
Address      Direction
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60011    100.100.100.1,
1048     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60011    100.100.100.2,
1034     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60012    100.100.100.1,
1049     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60012    100.100.100.2,
1035     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60013    100.100.100.1,
1050     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60013    100.100.100.2,
1036     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60014    100.100.100.1,
1051     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60014    100.100.100.2,
1037     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60015    100.100.100.1,
1052     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60015    100.100.100.2,
1038     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60016    100.100.100.1,
1053     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60016    100.100.100.2,
1039     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60017    100.100.100.1,
1054     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60017    100.100.100.2,
1040     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60018    100.100.100.1,
1024     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60018    100.100.100.2,
1041     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60019    100.100.100.1,
1025     Both
ppp-0/0/17/72339069014638604     tcp        11.100.128.25, 60019    100.100.100.2,
1042     Both
ipoe-0/0/18/216454257090494481   tcp        11.100.128.24, 60020    100.100.100.1,
1026     Both
```

Example for address translation rule summary. The example summary shows the total number of NAT rules for TCP flows.

```
supervisor@rtbrick.net: cfg> show address-translation rule summary
Instance: default
  Source     Protocol      Rules
  Dynamic    tcp              62
  Total Dynamic Rules         62
```

Example for address translation rules for subscribers.

```
supervisor@rtbrick: cfg> show address-translation rule
Instance: vrf1
User                             Protocol   Original Address        Translated
Address      Direction
ipoe-0/1/21/216454257090494746   tcp        11.100.129.0, 65001
100.100.100.101, 36874  Both
ipoe-0/1/21/216454257090494975   tcp        11.100.130.0, 65001
100.100.100.101, 60170  Both
ipoe-0/1/21/216454257090495232   tcp        11.100.131.0, 65001
100.100.101.101, 38730  Both
ppp-0/1/20/72339069014638594     tcp        10.100.128.1, 65001
100.100.100.100, 14914  Both
ipoe-0/1/21/216454257090494501   tcp        11.100.128.1, 65001
100.100.100.100, 2122   Both
ppp-0/1/20/72339069014638722     tcp        10.100.129.1, 65001
```

```
100.100.100.100, 19924  Both
ipoe-0/1/21/216454257090494747   tcp        11.100.129.1, 65001
100.100.100.100, 36746  Both
```

The example summary shows the total number of NAT rules for other protocol flows:

```
supervisor@rtbrick.net: cfg> show address-translation rule
Instance: vrf1
User                             Protocol   Original Address        Translated
Address      Direction
ipoe-0/1/30/216454257090494465   icmp       10.100.128.1, 7684
100.100.100.100, 1024   Both
```

NAT Trap Statistics

Syntax:

show address-translation trap statistics

This command displays the information of packets trapped in the NAT daemon.

Example for TCP flow stats

```
supervisor@rtbrick: cfg> show address-translation trap statistics
Protocol   Flags                      Status             Count
tcp        -|syn|-|-|-|-|-|-|-        Success           152958
tcp        -|syn|-|-|-|-|-|-|-        Failure            23186
Total Successful Traps:                                 152958
Total Failed Traps:                                      23186
```

Example for other flow (ICMP) stats:

```
supervisor@rtbrick>ufi09.q2c.u19.r4.nbg.rtbrick.net: cfg> show address-translation
trap statistics
Protocol   Flags                      Status             Count
other      -|-|-|-|-|-|-|-|-          Success               50
Total Successful Traps:                                     50
Total Failed Traps:                                          0
```

NAT Error Details

This command displays NAT error information.

Syntax:

show address-translation error

Example for address translation platform resource information.

```
supervisor@rtbrick.net: cfg> show address-translation error
INSTANCE - Packet Table
    Counter        : 4
    Current Update : Thu May 02 10:05:52 GMT +0000 2024
    Last Update    : Thu May 02 10:05:52 GMT +0000 2024
POOL_CFG - Pool Flush
    Counter        : 1
    Current Update : Thu May 02 10:05:43 GMT +0000 2024
    Last Update    : Thu May 02 10:05:43 GMT +0000 2024
```

Clear NAT Errors

Syntax

clear address-translation errors

This lear command resets the existing statistics for errors.

5.4. RBFS Carrier-Grade Network Address Translation Appliance

5.4.1. CGNAT Appliance Overview

Carrier-Grade Network Address Translation (CGNAT) appliance is a technology solution that addresses the IPv4 exhaustion challenge of service providers. The solution helps to use private network addresses and to limit the use of publicly routable IPv4 addresses significantly.

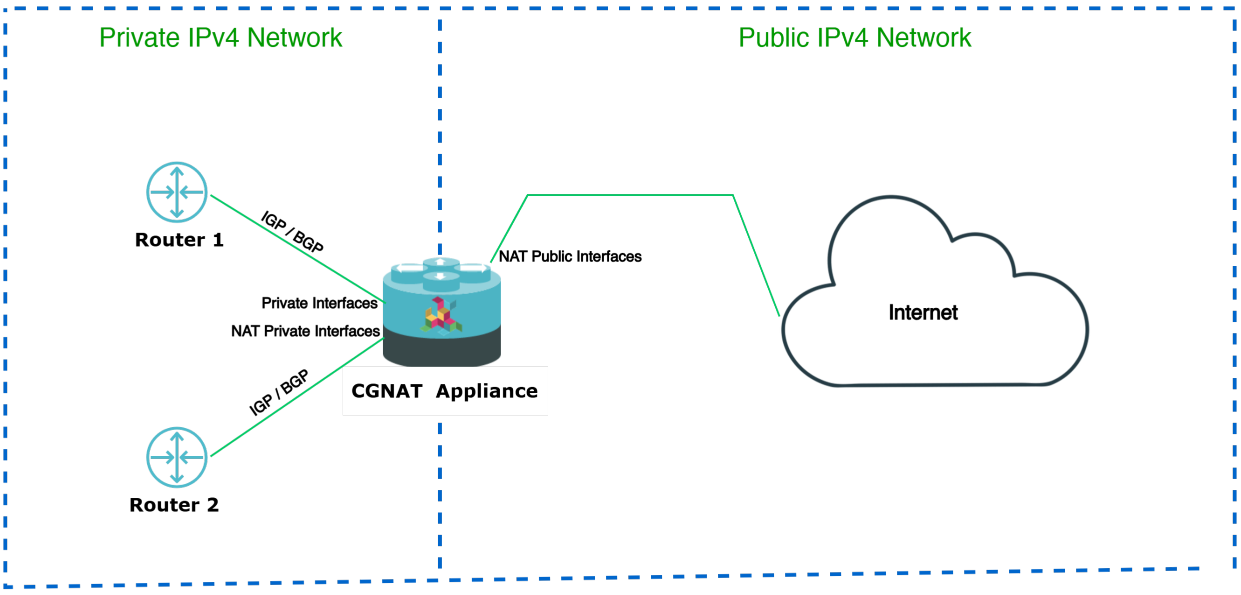
The main features of the CGNAT appliance are:

- Private Network Routers, such as BNGs, establish reachability within private networks and direct upstream traffic (from the private network to the internet) through a centralized CGNAT appliance. The CGNAT appliance is configured with NAT profiles, public pools, and separate private/public interfaces.
- Downstream reachability is achieved via aggregate NAT routes programmed by the CGNAT appliance and redistributed to BGP peers.
- Upstream reachability involves routers (Router 1 / Router 2 as shown in the diagram below) advertising private pool aggregate routes to the CGNAT

appliance.

- Traffic received at CGNAT private interfaces undergoes NAT treatment (SNAT).
- NAT Traffic received at CGNAT public interfaces undergoes NAT treatment(DNAT) and forwarded to the destination, while non-NAT IP traffic is forwarded seamlessly.

The following diagram illustrates a high-level view of CGNAT appliance.



There are three basic concepts in address translation:

Static NAT

Static NAT associates one private IP address, IP-protocol and source-L4-port with public IP address, IP protocol and source-L4-port. This is a one-to-one mapping and you must manually define the mapping between a private and public address and port (UDP/TCP).

Dynamic NAT

In Dynamic NAT, internal private IPv4 addresses and ports are mapped with a public IPv4 address and port, and this happens dynamically. The router dynamically picks an address from the address pool that is not currently assigned.

NAT444

CGNAT appliance supports NAT444. NAT 444 refers to three sets of IPv4 addresses: Customer private, ISP private, and public Internet. Network address translation occurs from customer private to ISP private, and then ISP private to public. The NAT444, also known as CGNAT, functionality maps IPv4 subscriber private addresses to IPv4 public addresses.

Benefits of CGNAT Appliance

CGNAT appliance allows service providers to provide services to a large number of their subscribers using a limited number of public IPv4 addresses. The benefits include:

Conserve Public IPv4 Addresses

The solution helps service providers conserve expensive public IPv4 addresses by enabling multiple subscribers to share a single public IPv4 address. With CGNAT, one public IPv4 address can manage hundreds of devices within the private network. It reduces the requirement of continually buying additional public IP addresses.

Enhanced Security

In addition to stretching the limited pool of public IPv4 addresses even further, CGNAT also provides significant security benefits. It helps to enhance network security by keeping the internal addressing private from the external network. It makes it difficult for attackers to target specific devices on the network by preventing attacks that target specific IP addresses.

Performance at Carrier-Grade

CGNAT appliance solution implements the functions of IPv4 address translation in the network of service providers. Service providers can deploy NAT in a way that allows multiple traffic flows on an L3 interface to share a single global IPv4 address and scale to several thousands of address translations. CGNAT operations are implemented at the forwarding plane (hardware level) to ensure optimal performance. This hardware-level implementation achieves high throughput with no performance impact and packet processing occurs without overburdening the CPU. The solution offers carrier-grade scalability with fast translation rates and a large number of IPv4 address and port number translations.

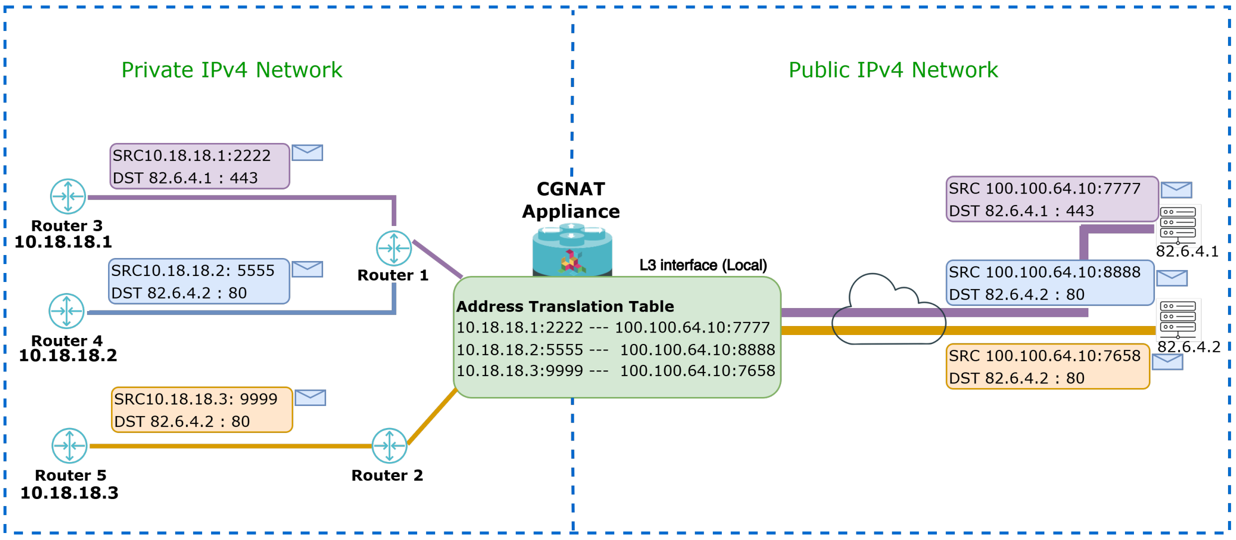
Understanding CGNAT Appliance Implementation

CGNAT appliance solution has been designed to support Port Address Translation, also known as Network Address Port Translation (NAPT), which has the greatest potential to conserve IPv4 addresses for service providers. NAPT is known as an effective method to allow multiple devices to connect to the Internet using a single public IPv4 address.

Network Address Port Translation

Network Address Port Translation is a dynamic NAT in which port numbers along with IP address are used to identify which traffic belongs to which private IP address. CGNAT appliance translates the source private IPv4 address and port number to a public source IPv4 address and unique port number. This allows multiple devices with different private IP addresses to use a single public IPv4 address. The unique port number ensures that the traffic is delivered to the correct device.

The following diagrams illustrate how CGNAT appliance works at a high level.



The diagram shows three subscribers using private IPv4 addresses (10.18.18.1, 10.18.18.2, and 10.18.18.3 send traffic to two different servers (82.6.4.1 and 82.6.4.2) on the public network.

CGNAT performs the address translation by replacing the source private IPv4 address with the public IPv4 address from the address pool and the source port number with a unique port number. After the address translation, the device forwards the traffic to the destination servers. After the translation, packets have a new source IPv4 address which is the public IPv4 and a unique port number as per the mapping in the address translation table.

CGNAT maintains a translation table with the entries. Entries are records of mapped private IPv4 addresses and port numbers with the public IPv4 addresses and port numbers.

The downstream traffic from the servers traverses to the CGNAT device. The packets coming from external hosts include the destination address as the (translated) public IPv4 address. CGNAT device performs reversal address translation for these packets as per the translation table mapping for the downstream traffic.

NAPT helps to significantly reduce the number of logs generated as it generates logs only during the allocation and release of each block of ports.

Deterministic NAT

CGNAT appliance offers support for deterministic NAT mode, which provides a consistent mapping of private IPv4 addresses with public IPv4 addresses and port ranges. This mode ensures a one-to-one mapping of private IPv4 addresses with public IPv4 addresses, allowing you to specify the private address and its matching public address and port range. The given private IPv4 address is always translated to the same public address.

In addition, deterministic NAT guarantees a predetermined and fixed translation for a given internal address with a public IPv4 and port combination, ensuring consistency and stability in the mapping. This feature is particularly useful for applications that involve security protocols, as well as for service providers who need to track private L3 interface traffic flows.

Furthermore, deterministic NAT significantly reduces address translation logs, as private IPv4 addresses are always mapped to public IPv4 addresses and port ranges.

Address Translation Table

Port mapping is a feature that allows multiple devices to share the same public IPv4 address with different port numbers. CGNAT generates a unique port number for each traffic flow on the L3 interface. Port mapping helps to determine the correct host among the many devices in the private network that use the same public IPv4 address.

CGNAT appliance maintains an address translation table that maps private IPv4 addresses and port numbers to their corresponding public IPv4 addresses and ports. Whenever an incoming packet arrives, CGNAT checks the translation table to see if a translation entry exists for that packet. If there’s an entry, the CGNAT replaces the source IPv4 address and port number in the packet header with the mapped public IPv4 address and port number.

When a device on the Internet sends packets downstream, the CGNAT software uses the translation table for address translation reversal. It replaces the destination public IPv4 address and port number in the packet header with the corresponding private IPv4 address and port number.

NAT IP Pools and Chaining

A NAT pool contains a range of multiple public IP addresses. You can create multiple pools and associate them with a NAT service. Pool chaining is a method in which you can associate one pool with another. For a pool, you can define 'next pool name'; so that when the pool gets exhausted with the IPv4 addresses, the next pool that is defined will take over.

Aging

Aging refers to the time that a translation entry exists or remains in the address translation table after it was last used.

The entries in the translation table have a finite lifespan as the software implements mechanisms to handle session timeouts.

When a host sends a packet to a destination, CGNAT translates the private IPv4 address and port to a public IPv4 address and port. This mapping is recorded as an entry in the address translation table. The software always looks up the translation table whenever it receives a packet to verify that any entry exists for the packet in the translation table. The software performs the address translation based on the recorded entries and its associated mappings for both inbound and outbound packets.

If there is no activity related to a specific mapping (for example, when there are no incoming or outgoing packets), the mapping will eventually be removed from the address translation table, once idle or unused entries are detected, entries removed from the address translation table to free up resources for the new upcoming traffic flows.

For TCP, idle or unused flows are typically detected by the receipt of TCP FIN-ACK packets, which indicate that both sides of the connection have finished sending data. Once detected, these flows are removed after a configurable aging timer expires.

For UDP, as it is a connectionless protocol, idle or unused flows are detected by periodically polling the flow traffic. As the number of UDP traffic flows increases, the idle flow detection mechanism also increases, ensuring that idle flows are removed in a timely manner. The aging of UDP flows is proportional to the number of traffic flows.

```
NOTE: The NAT ageing timeout is 200 seconds for TCP and UDP traffic. The actual
time taken to flush aged-out NAT rules depends on the number of NAT rules and the
number of counters.
```

Logging

CGNAT appliance provides traffic monitoring capabilities to track and log network activities. The solution offers a flexible logging mechanism that can store information such as port numbers, time, destination, and address translation details. You can enable logging for CGNAT operations.

CGNAT Handling of ICMP Messages

The NAT-enabled RBFS devices can perform network address translation for ICMP echo request and reply messages. Use the ping command (ICMP Echo Request, Echo Reply) to check internet connectivity to external internet addresses when NAT is enabled. This feature is enabled by default.

CGNAT Handling of TCP Control Messages

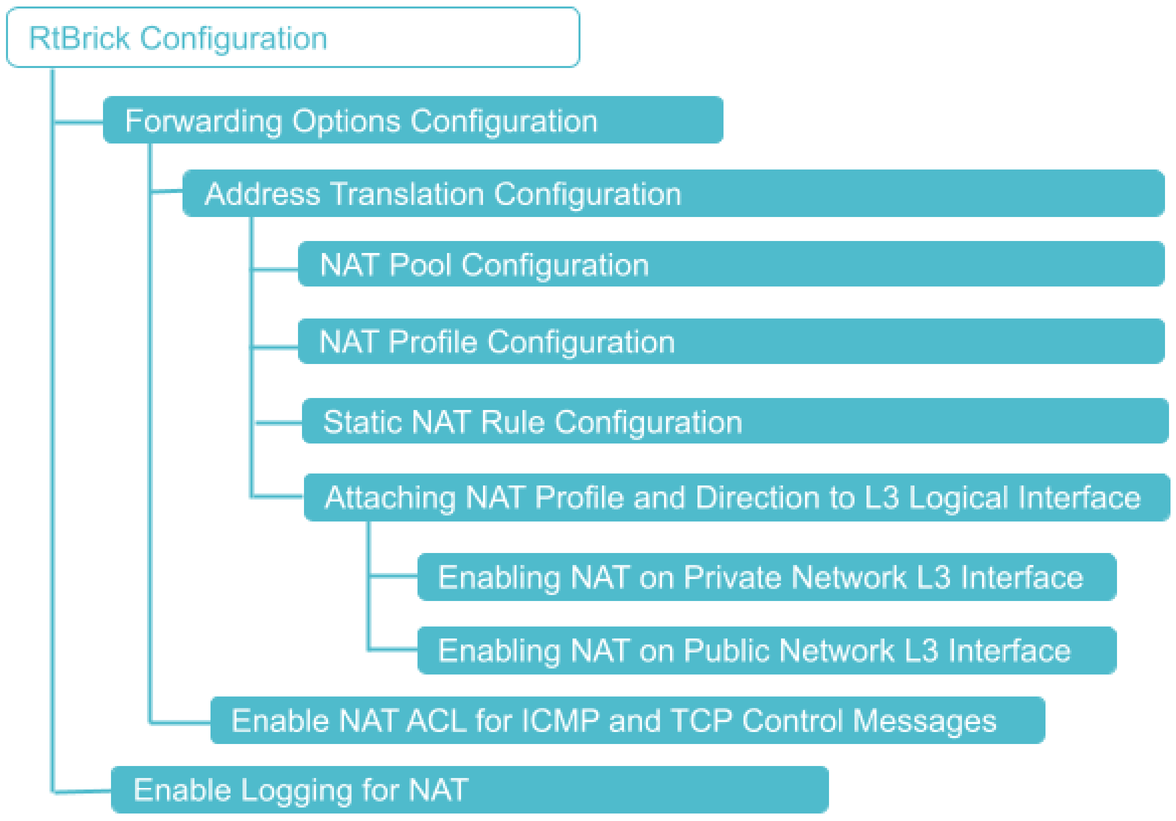
RBFS with NAT supports address translation for TCP control messages using ACLs. It includes the TCP control messages such as FIN, FIN-ACK, RST, RST-ACK, SYN, and SYN-ACK. This feature is enabled by default.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

5.4.2. CGNAT Appliance Configuration

Configuration Hierarchy


RtBrick Configuration
Forwarding Options Configuration
Address Translation Configuration
NAT Pool Configuration
NAT Profile Configuration
Static NAT Rule Configuration
Attaching NAT Profile and Direction to L3 Logical Interface
Enabling NAT on Private Network L3 Interface
Enabling NAT on Public Network L3 Interface
Enable NAT ACL for ICMP and TCP Control Messages
Enable Logging for NAT

CGNAT Appliance Configuration

You must perform the following tasks to configure CGNAT appliance.

1. Configure NAT Pool
2. Configure NAT Profile
3. Enable NAT on an Instance
4. Attaching NAT Profile and Direction to L3 Logical Interface
5. Enable NAT on Private Network L3 Interface
6. Enable NAT on Public Network L3 Interface
7. Configure NAT Rule
8. Enable Logging for NAT
9. Enable NAT ACL Support for ICMP and TCP Control Messages

Configuration Syntax and Commands

The following sections describe the CGNAT appliance configuration syntax and commands.

NAT Pool Configuration

A NAT IP address pool includes a set of public IPv4 addresses that are used for network address translation. You can create multiple public IPv4 address pools and one pool includes a range of public IPv4 addresses. These pools allocate public IPv4 addresses to subscribers during address translation. While configuring a pool, you can define the group of public IPv4 addresses belonging to that pool by specifying the lowest and highest IP addresses.

The system allows you to create multiple pools and define the association among them. You can define the 'next-pool-name' that takes over when the current pool gets exhausted with the IPv4 addresses. When one pool gets exhausted, the next pool takes over and starts serving the IP addresses to subscribers when the address translation occurs.

Syntax:

set forwarding-options address-translation pool <pool-name> <attribute> <value>

Attribute	Description
<pool-name>	Specify the name of the address pool.
ipv4-address	Specify both the highest and lowest IPv4 addresses in the range of IPv4 addresses for the pool.
ipv4-address high	Specify the highest IPv4 address in the address pool. You must specify the highest IP address in the range of IP addresses.
ipv4-address low	Specify the lowest IPv4 address in the address pool. You must specify the lowest IP address in the range of IP addresses.
next-pool-name	Specify the name of the next address pool that is to be used when the current address pool is allocated completely.

Example Configuration:

The following commands configure nataddr_pool1 as the NAT pool and nataddr_pool2 as the next pool.

It indicates NAT pool nataddr_pool1 contains a range of public IPv4 addresses from 100.100.100.1 to 100.100.100.5.

When the pool nataddr_pool1 has fully allocated its IPv4 addresses, the next pool named nataddr_pool2 will start allocating IPv4 addresses from its pool. The pool nataddr_pool2 includes a rage of IPv4 address from 100.100.101.1 to 100.100.101.150.

```
set forwarding-options address-translation pool nataddr_pool1
set forwarding-options address-translation pool nataddr_pool1 next-pool-name
nataddr_pool2
set forwarding-options address-translation pool nataddr_pool1 ipv4-address low
100.100.100.1
set forwarding-options address-translation pool nataddr_pool1 ipv4-address high
100.100.100.5
set forwarding-options address-translation pool nataddr_pool2 ipv4-address low
100.100.101.1
set forwarding-options address-translation pool nataddr_pool2 ipv4-address high
100.100.101.150
```

Example Configuration:

```
supervisor@rtbrick: cfg> show config forwarding-options address-translation pool
{
  "rtbrick-config:pool": [
    {
      "pool-name": "nataddr_pool1",
      "next-pool-name": "nataddr_pool2",
      "ipv4-address": {
        "low": "100.100.100.1",
        "high": "100.100.100.5"
      }
    },
    {
      "pool-name": "nataddr_pool2",
      "ipv4-address": {
        "low": "100.100.101.1",
        "high": "100.100.101.150"
      }
    }
  ]
}
```

Configuring NAT Profile

A NAT profile defines how the NAT device has to perform the IPv4 address translation. NAT profile allows you to define an instance, IPv4 address pools, maximum number of translations, and mapping a particular internal IPv4 address with a particular external IPv4 address for a deterministic address translation.

You can create NAT profile for an RBFS instance using the 'instance' option. Also, you can define the TCP or UDP traffic type for the profile.

A single NAT profile can be attached to subscribers across different instances. This means that regardless of which instance a subscriber belongs to, they can share the same NAT profile. For example, subscribers in different routing instances can be managed using the same NAT settings.

Subscribers within the same instance can be assigned different NAT profiles. Different groups of subscribers or services within the same instance can have tailored NAT configurations.

Syntax:

set forwarding-options address-translation profile <profile-name> <attribute> <value>

Attribute	Description
<profile-name>	Specify the NAT profile name.
deterministic [true]	Specify deterministic as true to enable deterministic NAT for the profile. Deterministic NAT allows subscribers always to connect with a single public IP.
instance	Specify the RBFS instance.
ip-protocol	Specify the protocol: TCP or UDP. Note: When using the tcp-control match attribute in a match rule, it is necessary to explicitly configure match ip-protocol for TCP. This ensures that the rule accurately identifies and processes TCP packets before applying any matching for TCP control flags.
ip-protocol ageing-timeout <ageing-timeout>	Specify the ageing time value for the protocol. Default, 120 seconds. Allowed range: 60 - 86400 seconds.

Attribute	Description
ip-protocol control-plane-ageing-timeout <ageing-timeout>	Specify the TCP control ageing timeout value. Default, 10 seconds. Allowed range: 5 - 120 seconds.
max-rules	Specify the maximum number of rules for address translations for a public IPv4 address and for an interface. The supported values for maximum address translation for an interface are 64, 128, 256, 512, 1024, 2048, 4096, 8192, and 16342.
pool	Specify the name of the public IP address pool.

The following commands configure the NAT profile named nat_profile1. The nat profile nat_profile1 is configured on the instanced vrf1 with a pool attached nataddr_pool1. Maximum rules are configured as 100 rules and the aging period is configured as 600 seconds for TCP traffic and 300 seconds for UDP traffic.

```
set forwarding-options address-translation profile nat_profile1
set forwarding-options address-translation profile nat_profile1 instance vrf1
set forwarding-options address-translation profile nat_profile1 pool nataddr_pool1
set forwarding-options address-translation profile nat_profile1 ip-protocol TCP
ageing-timeout 600
set forwarding-options address-translation profile nat_profile1 ip-protocol UDP
ageing-timeout 300
set forwarding-options address-translation profile nat_profile1 ip-protocol other
ageing-timeout 300
```

Example Configuration:

```
supervisor@rtbrick>multiservice-edge1.rtbrick.net: cfg> show config forwarding-
options address-translation profile
{
  "rtbrick-config:profile": [
    {
      "profile": "nat_profile1",
      "instance": "vrf1",
      "pool": "nataddr_pool1",
      "ip-protocol": {
        "TCP": {
          "ageing-timeout": 600
        },
        "UDP": {
          "ageing-timeout": 300
        },
        "other": {
          "ageing-timeout": 300
        }
      }
    }
  ]
}
```

Enable NAT on an Instance

You can enable address translation for a specific routing instance such as VRF or virtual router rather than globally across the entire router.

Syntax:

set instance <instance-name> **address-translation true**

Attribute	Description
<instance-name>	Name of the instance.

The following command is used to enable address translation for the instance named 'vrf1'.

```
set instance vrf1 address-translation true
```

```
supervisor@rtbrick.net: op> show config instance vrf1 address-translation
{
  "rtbrick-config:address-translation": "true"
}
```

Attaching NAT Profile and Direction to L3 Logical Interface

Enabling NAT on a Private Network L3 Interface

To enable NAT on a private network L3 interface, you need to configure the direction of the network interface as the "local".

Syntax:

set interface <interface-name> **unit** <unit-id> **address-translation** <attribute> <value>

Attribute	Description
<interface-name>	Name of the interface. Example: ifp-0/0/1.

Attribute	Description
unit <unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface.
profile <profile-name>	Specify the NAT profile name.
direction local	Indicates the NAT configuration on a private network.

The following command attaches the NAT profile nat_profile in the local direction to an L3 logical interface.

```
supervisor@rtbrick>multiservice-edge1.rtbrick.net: cfg> show config set interface
ifp-0/0/1
set interface ifp-0/0/1
set interface ifp-0/0/1 unit 100
set interface ifp-0/0/1 unit 100 address-translation
set interface ifp-0/0/1 unit 100 address-translation direction local
set interface ifp-0/0/1 unit 100 address-translation profile nat_profile
```

Example Configuration:

```
supervisor@rtbrick>multiservice-edge1.rtbrick.net: cfg> show config  interface
ifp-0/0/1
{
  "rtbrick-config:interface": [
    {
      "name": "ifp-0/0/1",
      "unit": [
        {
          "unit-id": 100,
          "address-translation": {
            "direction": "local",
            "profile": "nat_profile"
          }
        }
      ]
    }
  ]
}
```

Enabling NAT on a Public Network L3 Interface

To enable NAT on a public network L3 interface, you need to configure the direction of the network interface as the "public".

Syntax:

set interface <interface-name> **unit** <unit-id> **address-translation** <attribute>

<value>

Attribute	Description
<interface-name>	Name of the interface.
<unit-id>	Configure the number of sub-interfaces under the physical interface.
direction public	Specify 'public' for the external interface.

The following commands configure the external interface ifp-0/1/64 for IPv4 address translation. 'Unit' logical identifier for this physical interface. Direction 'public' shows the configuration on external interface for address translation.

```
set interface ifp-0/1/64 unit 100
set interface ifp-0/1/64 unit 100 address-translation
set interface ifp-0/1/64 unit 100 address-translation direction public
```

```
supervisor@rtbrick.net: cfg> show config interface ifp-0/1/64 unit 100
{
  "rtbrick-config:unit": [
    {
      "unit-id": 100,
      "address-translation": {
        "direction": "public"
      }
    }
  ]
}
```

NAT Rule Configuration

You can define NAT rules only for static NAT. A NAT rule defines a match condition and a corresponding action. After you specify NAT rules, each packet is matched with each NAT rule. If a packet matches the condition specified in a rule, then the action corresponding to that match occurs. Match rules govern how the translation of private IPv4 addresses to public IPv4 addresses is performed.

With NAT rules, you can define how address translation is applied to traffic, and how to handle various protocols and data traffic, such as TCP and UDP, to ensure proper address translation and the mappings of private addresses to public addresses.

Rules also define how to handle inbound and outbound traffic, different protocols, and data traffic such as TCP and UDP for ensuring the proper address translation of traffic.

A maximum of 16,384 NAT rules on NAT profile is allowed.

Syntax:

set forwarding-options address-translation rule <rule-name> <attribute> <value>

Attribute	Description
<rule-name>	Specify the name of the rule.
ordinal <ordinal-value>	Specify the ordinal value. An ordinal value is a numerical representation that indicates its relative position or order.
ordinal <ordinal-value> instance	Specify the RBFS instance name.
ordinal <ordinal-value> ip-protocol [tcp/udp]	Specify the IP protocol, TCP or UDP.
ordinal <ordinal-value> local [ipv4-address/port]	Specify the private IPv4 address or port number that needs to be translated.
ordinal <ordinal-value> public [ipv4-address/port]	Specify the public IPv4 address. This public IP will be mapped with the private IP in the translation table.

Enable Logging for NAT

You can optionally enable logging for CGNAT appliance operations.

All RBFS logs and related information is available in the *RBFS Logging User Guide*. For the list of RBFS logs, see Log Reference.

set log bd <name> <options>

Attribute	Description
level	Specify the log level.
module	Specify the log module.

Attribute	Description
plugin-alias	Specify the plugin-alias URL. Plugin-alias is an external logging host server to which you can export logs. For example, Graylog.

The following commands configure logging for NAT module natd with a log level 'debug'.

```
set log bd natd
set log bd natd module nat
set log bd natd module nat level debug
```

```
supervisor@rtbrick.net: cfg> show config log bd natd
{
  "rtbrick-config:bd": [
    {
      "bd-name": "natd",
      "module": [
        {
          "module-name": "nat",
          "level": "debug"
        }
      ]
    }
  ]
}
```

Enabling NAT ACL Support for ICMP and TCP Control Messages

You can enable Access Control Lists support for ICMP and TCP Control messages.

Enabling NAT Support for ICMP Using ACLs

The NAT ACLs for ICMP packets allows address translation for ICMP request and response packets through the use of Access Control Lists (ACLs). This enhances the handling of ICMP traffic by providing precise control over NAT behavior for ICMP packets.

Setting up the NAT ACLs for ICMP will be handled by the NAT application with the 25.1.1 release, eliminating the need for manual configuration.

set forwarding-options acl l3v4 rule rule-name> ordinal <ordinal-number> <attribute > <value>

Attribute	Description
<rule-name>	Specify the name of the rule.
ordinal <ordinal-value>	Specify the ordinal value. An ordinal value is a numerical representation that indicates its relative position or order.
match	Define the conditions under which the rule will be applied. These conditions determine which packets the rule should act on.
action	Defines what happens to packets that match the rule’s criteria.
priority	Determines the priority of the rule when multiple rules with the same ordinal value are evaluated.
direction	The rule applies to ingress or egress traffic.

Example commands:

```
set forwarding-options acl l3v4 rule NAT_TRAP
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 match ip-protocol icmp
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 match destination-ipv4-
prefix-list PREFIXES_NAT_PUBLIC
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1001 priority 1001

set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 match source-ipv4-
prefix 100.64.0.0/10
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 match ip-protocol icmp
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1002 priority 1002
```

Example:

```
supervisor@rtbrick>rtbrick.net: cfg> show config forwarding-options acl l3v4
{
  "rtbrick-config:l3v4": {
    "rule": [
      {
        "rule-name": "NAT_TRAP",
        "ordinal": [
          {
            "ordinal-value": 1001,
            "match": {
              "direction": "ingress",
              "ip-protocol": "icmp",
              "destination-ipv4-prefix-list": "PREFIXES_NAT_PUBLIC"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1001
          },
          {
            "ordinal-value": 1002,
            "match": {
              "direction": "ingress",
              "source-ipv4-prefix": "100.64.0.0/10",
              "ip-protocol": "icmp"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1002
          }
        ]
      }
    ]
      }
    }
```

The preceding configuration defines Layer 3 IPv4 Access Control List (ACL) rules for managing ICMP traffic. The rule with the ordinal value 1001 is designed to redirect ICMP traffic for Network Address Translation (NAT) processing based on specific criteria. The ACL rule is named NAT_TRAP.

The matching criteria for this rule include the direction of traffic, which is set to ingress. This means the rule applies only to incoming traffic. The specified protocol is ICMP, indicating that the rule is applicable to ICMP packets.

The 'destination-ipv4-prefix-list' is designated as PREFIXES_NAT_PUBLIC, which is a predefined list containing a range of IPv4 addresses. The rule specifically pertains to ICMP packets whose destination IP is included in the PREFIXES_NAT_PUBLIC list. These rules ensure that ICMP packets requiring NAT handling are identified and appropriately redirected to the NAT process. When an ICMP packet that meets the conditions (ingress, ICMP, and a destination in PREFIXES_NAT_PUBLIC) is detected, it is redirected (or punted) to the NAT process for further handling.

Enabling NAT Support for TCP Control Messages Using ACL

This configuration enables address translation on TCP control messages through ACLs. The feature supports control messages such as FIN, SYN, and RST. It applies functionality to TCP control messages such as fin-ack and rst-ack.

Syntax:

set forwarding-options acl l3v4 rule <rune-name> **ordinal** <ordinal-number> **match tcp-control** [fin-ack | rst-ack]

Attribute	Description
<rule-name>	Specify the name of the rule.
ordinal <ordinal-value>	Specify the ordinal value. An ordinal value is a numerical representation that indicates its relative position or order.
match	Define the conditions under which the rule will be applied. These conditions determine which packets the rule should act on.
action	Defines what happens to packets that match the rule's criteria.
priority	Determines the priority of the rule when multiple rules with the same ordinal value are evaluated.
direction	The rule applies to ingress or egress traffic.
tcp-control <fin-ack>	Matches TCP packets with the FIN-ACK control flag set. When using the tcp-control match attribute in a match rule, it is necessary to explicitly configure match ip-protocol for TCP. This ensures that the rule accurately identifies and processes TCP packets before applying any matching for TCP control flags.
destination-ipv4-prefix-list	Name of the prefix list. The list includes predefined destination IP addresses.

Example commands:

```
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 match tcp-control fin-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 match destination-ipv4-
prefix-list PREFIXES_NAT_PUBLIC
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1003 priority 1003
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 match source-ipv4-
prefix 100.64.0.0/10
```

```
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 match tcp-control fin-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1004 priority 1004
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 match tcp-control rst-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 match destination-ipv4-
prefix-list PREFIXES_NAT_PUBLIC
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1005 priority 1005
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 match direction ingress
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 match source-ipv4-
prefix 100.64.0.0/10
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 match tcp-control rst-
ack
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 action trap nat
set forwarding-options acl l3v4 rule NAT_TRAP ordinal 1006 priority 1006
```

Example:

```
supervisor@rtbrick>rtbrick.net: cfg> show config forwarding-options acl l3v4
{
  "rtbrick-config:l3v4": {
    "rule": [
      {
        "rule-name": "NAT_TRAP",
        "ordinal": [
          {
            "ordinal-value": 1003,
            "match": {
              "direction": "ingress",
              "tcp-control": "fin-ack",
              "destination-ipv4-prefix-list": "PREFIXES_NAT_PUBLIC"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1003
          },
          {
            "ordinal-value": 1004,
            "match": {
              "direction": "ingress",
              "source-ipv4-prefix": "100.64.0.0/10",
              "tcp-control": "fin-ack"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1004
          },
          {
            "ordinal-value": 1005,
            "match": {
              "direction": "ingress",
              "tcp-control": "rst-ack",
              "destination-ipv4-prefix-list": "PREFIXES_NAT_PUBLIC"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1005
          },
          {
            "ordinal-value": 1006,
            "match": {
              "direction": "ingress",
              "source-ipv4-prefix": "100.64.0.0/10",
              "tcp-control": "rst-ack"
            },
            "action": {
              "trap": "nat"
            },
            "priority": 1006
          }
        ]
      }
    ]
  }
}
```

The preceding configuration defines ACL for Layer 3 IPv4 traffic. The ACL is named NAT_TRAP and consists of multiple rules that match specific traffic patterns and apply a 'trap' action to punt the traffic for further NAT processing. Each rule is identified by an ordinal value.

Match Criteria includes direction which is specified as ingress. All rules are applied to ingress traffic. Protocol is defined as TCP traffic with specific control flags: fin-ack or rst-ack.

In the configuration:

- Ordinal 1003 matches TCP traffic with the fin-ack flag and a destination IP in the PREFIXES_NAT_PUBLIC list.
- Ordinal 1004 matches TCP traffic with the fin-ack flag and a source IP in the 100.64.0.0/10 range.
- Ordinal 1005 matches TCP traffic with the rst-ack flag and a destination IP in the PREFIXES_NAT_PUBLIC list.
- Ordinal 1006 matches TCP traffic with the rst-ack flag and a source IP in the 100.64.0.0/10 range.

5.4.3. CGNAT Appliance Operational Commands

CGNAT Appliance Show Commands

The CGNAT operational commands provide detailed information about the address translation operations.

NAT Pool Information

This command displays information about IPv4 address pool for a user.

Syntax:

show address-translation allocation <options>

Option	Description
pool	Specify the pool name.
user	Specify the user name.

Example for NAT pool allocation.

```
supervisor@rtbrick.net: cfg> show address-translation allocation pool
Pool: pool1
  Profile: nat_profile1
  Instance: vrf1
  Allocated: 100.00%
Pool: pool2
  Profile: nat_profile1
  Instance: vrf1
  Allocated: 100.00%
```

NAT Rule Details

Syntax:

show address-translation rule <options>

Option	Description
-	Without any option, the command displays the information for all rules.
instance	Displays NAT rule information for the specified instance.

Option	Description
summary	Displays summary of the information for the NAT rule.
user	Displays user information for the NAT rule.

Example for address translation rules for subscribers.

```
supervisor@rtbrick: cfg> show address-translation rule
Instance: vrf1
User                              Protocol   Original Address        Translated
Address      Direction
ifl-0/1/6/100                     tcp        11.100.129.0, 60011     100.200.200.6,
2825     Both
ifl-0/1/6/100                     tcp        11.100.130.0, 60011     100.200.200.9,
2027     Both
ifl-0/1/6/100                     tcp        11.100.128.1, 60011
100.200.200.39, 3283    Both
ifl-0/1/6/100                     tcp        11.100.129.1, 60011     100.200.200.4,
2828     Both
ifl-0/1/6/100                     tcp        11.100.131.1, 60011
100.200.200.32, 1260    Both
ifl-0/1/6/100                     tcp        11.100.129.3, 60011
100.200.200.46, 2796    Both
ifl-0/1/6/100                     tcp        11.100.128.4, 60011
100.200.200.30, 3328    Both
ifl-0/1/6/100                     tcp        11.100.130.4, 60011     100.200.200.3,
1945     Both
ifl-0/1/6/100                     tcp        11.100.129.5, 60011
100.200.200.39, 2796    Both
ifl-0/1/6/100                     tcp        11.100.130.5, 60011
100.200.200.45, 1934    Both
ifl-0/1/6/100                     tcp        11.100.128.6, 60011
100.200.200.25, 3304    Both
ifl-0/1/6/100                     tcp        11.100.129.7, 60011
100.200.200.36, 2650    Both
ifl-0/1/6/100                     tcp        11.100.130.7, 60011
100.200.200.31, 1982    Both
ifl-0/1/6/100                     tcp        11.100.128.8, 60011
100.200.200.18, 3294    Both
ifl-0/1/6/100                     tcp        11.100.130.8, 60011     100.200.200.5,
3651     Both
ifl-0/1/6/100                     udp        11.100.129.0, 60011     100.200.200.6,
2825     Both
ifl-0/1/6/100                     udp        11.100.130.0, 60011     100.200.200.9,
2027     Both
ifl-0/1/6/100                     udp        11.100.128.1, 60011
100.200.200.39, 3283    Both
ifl-0/1/6/100                     udp        11.100.129.1, 60011     100.200.200.4,
2828     Both
ifl-0/1/6/100                     udp        11.100.131.1, 60011
100.200.200.32, 1260    Both
ifl-0/1/6/100                     udp        11.100.129.3, 60011
100.200.200.46, 2796    Both
ifl-0/1/6/100                     udp        11.100.128.4, 60011
100.200.200.30, 3328    Both
ifl-0/1/6/100                     udp        11.100.130.4, 60011     100.200.200.3,
1945     Both
```

```
ifl-0/1/6/100                    udp        11.100.129.5, 60011
100.200.200.39, 2796    Both
ifl-0/1/6/100                    udp        11.100.130.5, 60011
100.200.200.45, 1934    Both
ifl-0/1/6/100                    udp        11.100.128.6, 60011
100.200.200.25, 3304    Both
ifl-0/1/6/100                    udp        11.100.129.7, 60011
100.200.200.36, 2650    Both
ifl-0/1/6/100                    udp        11.100.130.7, 60011
100.200.200.31, 1982    Both
ifl-0/1/6/100                    udp        11.100.128.8, 60011
100.200.200.18, 3294    Both
ifl-0/1/6/100                    udp        11.100.130.8, 60011     100.200.200.5,
3651     Both
<...>
```

The example summary shows the total number of NAT rules for other protocol flows:

```
supervisor@rtbrick.net: cfg> show address-translation rule
Instance: vrf1
User                             Protocol   Original Address        Translated
Address      Direction
ifl-0/1/6/100                    icmp       10.100.128.1, 7684
100.100.100.100, 1024   Both
```

NAT Trap Statistics

Syntax:

show address-translation trap statistics

This command displays the information of packets trapped in the NAT daemon.

Example for TCP flow stats

```
supervisor@rtbrick: cfg> show address-translation trap statistics
Protocol   Flags                     Status            Count
tcp        -|syn|-|-|-|-|-|-|-       Success          152958
tcp        -|syn|-|-|-|-|-|-|-       Failure           23186
Total Successful Traps:                               152958
Total Failed Traps:                                    23186
```

Example for TCP, UDP, ICMP(other) flow stats

```
supervisor@rtbrick: cfg> show address-translation trap statistics
Protocol   Flags                     Status            Count
tcp        -|syn|-|-|-|-|-|-|-       Success              60
tcp        fin|-|-|-|ack|-|-|-|-     Success               4
tcp        -|syn|-|-|ack|-|-|-|-     Failure              12
```

```
udp         -|-|-|-|-|-|-|-|-         Success             182
udp         -|-|-|-|-|-|-|-|-         Failure             326
other       -|-|-|-|-|-|-|-|-         Success            5292
Total Successful Traps:                                  5538
Total Failed Traps:                                       338
```

NAT Error Details

This command displays NAT error information.

Syntax:

show address-translation error

Example for address translation platform resource information.

```
supervisor@rtbrick.net: cfg> show address-translation error
INSTANCE - Packet Table
    Counter        : 4
    Current Update : Thu May 02 10:05:52 GMT +0000 2024
    Last Update    : Thu May 02 10:05:52 GMT +0000 2024
POOL_CFG - Pool Flush
    Counter        : 1
    Current Update : Thu May 02 10:05:43 GMT +0000 2024
    Last Update    : Thu May 02 10:05:43 GMT +0000 2024
```

Clear NAT Errors

Syntax:

clear address-translation errors

This lear command resets the existing statistics for errors.

5.5. RBFS HTTP Redirect Service

5.5.1. RBFS HTTP Redirect Service Overview

This document provides information about RBFS HTTP Redirect Service. For enabling HTTP Redirect service, it is required to create, associate, and apply subscriber filters (ACLs) for the subscribers. For information about subscriber filters and how to configure these filters, refer to the RBFS Subscriber Filters User Guide.

RBFS HTTP Redirect service allows network service providers to intercept and redirect HTTP request traffic from subscribers to a designated captive portal instead of the original destination. This service is useful in a multitude of use cases, ranging from subscriber re-authentication to enforcing acceptance of network usage policies.

This captive portal is a webpage where the redirected subscribers are landed up to fulfill certain actions or conditions before they are granted broader access to the network resources. There are various reasons why captive portals can be set up, such as the following:

- Accept the terms of service.
- Receive and manage HTTP requests to unauthorized web resources.
- Present a web page that requires the completion of certain actions from the subscriber.
- Serve commercial communication or network usage policy messages.

By implementing the RBFS HTTP Redirect Service, network service providers can efficiently manage user access and enforce compliance with network regulations and policies, ultimately enhancing the overall security and user experience within their network environment.

Based on the applied filters, RBFS performs three actions which are accept, drop, and redirect. The action Accept allows the subscribers to access the network resource that they request. The Drop action restricts the subscriber from accessing the network resource. Finally, the Redirect action, if enabled HTTP redirect service, takes the subscriber to a different portal where the service provider wants to fulfill certain actions by the subscriber before accessing the network resource.

In addition to RBFS Subscriber Filters, a redirect action is supported by RBFS through the utilization of Ascend Data Filters (ADF), as described in the RBFS RADIUS Services Guide.

The RBFS HTTP Redirect Service together with the Subscriber Filters empowers network service providers to intercept and redirect HTTP request traffic from subscribers, guiding it towards a designated captive portal instead of its original destination.

How RBFS HTTP Redirect Service Works

HTTP requests from subscribers to any destination are intercepted by the RBFS HTTP Redirect service, if the subscriber filter rules are applied to the subscriber. In response to the request from the subscriber to access the network, the HTTP Redirect service provides the HTTP status code 302 along with the new URL to guide the subscriber to the new destination. To make it possible, the RBFS Subscriber Filters are employed, enabling the service to decide which requests should be redirected and which requests should be passed directly without redirection. In addition to RBFS Subscriber Filters, a redirect action is supported by RBFS through the utilization of Ascend Data Filters (ADF), as described in the RBFS RADIUS Services Guide.

Enabling HTTP Redirect Service Using RADIUS ADF

Both the RBFS Subscriber Filters and the HTTP Redirect Service can be dynamically enabled, disabled, and updated through RADIUS access-accept and CoA requests without requiring the re-establishment of the subscriber session. This flexibility allows network administrators to efficiently manage and modify these services as needed without disrupting subscriber connectivity.

RBFS provides a set of vendor-specific attributes to control the RBFS Subscriber Filters and the HTTP Redirect Service. The attributes include:

- VSA 26-50058-75 - RtBrick-HTTP-Redirect-URL
- VSA 26-50058-76 - RtBrick-IPv4-ACL-IN
- VSA 26-50058-77 - RtBrick-IPv4-ACL-OUT
- VSA 26-50058-78 - RtBrick-IPv6-ACL-IN
- VSA 26-50058-79 - RtBrick-IPv6-ACL-OUT

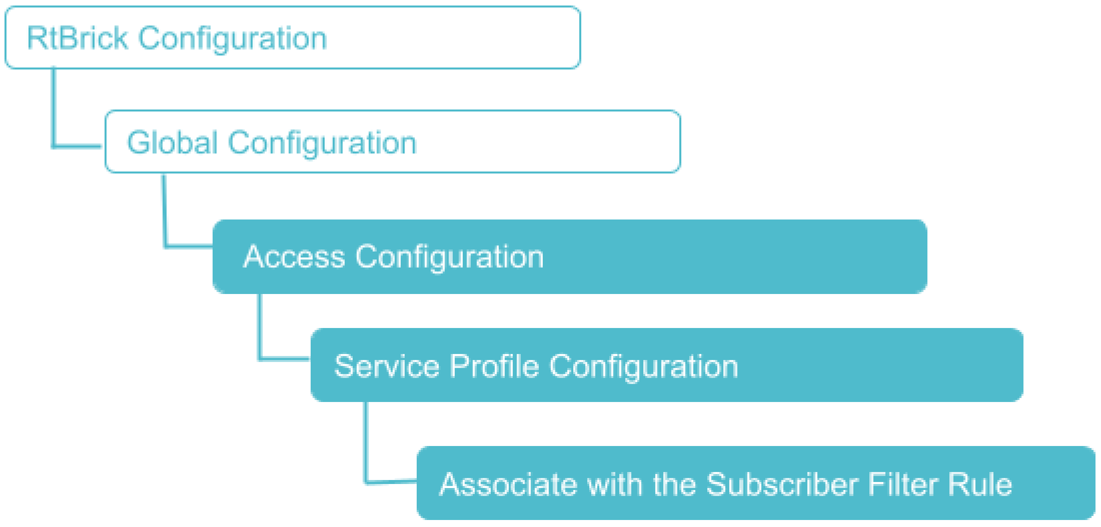
For more information about these attributes, see RBFS RADIUS Services Guide.

Limitations

The HTTP Redirect Service is currently supported only for IPoE subscribers. HTTP Redirect service can redirect only HTTP traffic. Typically, the majority of web requests are in HTTPS format, which cannot be redirected. Due to this limitation, most end-user devices, including PCs, smartphones, and tablets, automatically attempt to access specific well-known URLs to search for captive portals immediately after establishing a network connection. Two common examples are http://connectivitycheck.gstatic.com (vendor-independent) and http://captive.apple.com (Apple devices).

5.5.2. HTTP Redirect Service Configuration

The configuration hierarchy for the HTTP Redirect service is illustrated in the diagram.


RtBrick Configuration
Global Configuration
Access Configuration
Service Profile Configuration
Associate with the Subscriber Filter Rule

Service Profile Configuration

Syntax:

set access service-profile profile-name <profile-name> <attribute> <value>

Attribute	Description
<profile-name>	Service profile name.
<http-redirect>	HTTP redirect service configuration.
<url>	HTTP redirect target URL.
<acl>	Subscriber ACL (filter) configuration.
<ipv4-acl-in>	IPv4 upstream ACL (ingress from subscriber).
<ipv4-acl-out>	IPv4 downstream ACL (egress to subscriber).
<ipv6-acl-in>	IPv6 upstream ACL (ingress from subscriber).
<ipv6-acl-out>	IPv6 downstream ACL (egress to subscriber).

The following example shows a service profile named HTTP, redirect URL as the redirect destination. The 'ipv4-acl-in' ACL is applied as the filter criteria or ACL rule for this redirection.

```
supervisor@router: cfg> show config access service-profile HTTP
{
  "rtbrick-config:service-profile": [
    {
      "profile-name": "HTTP",
      "http-redirect": {
        "url": "https://www.rtbrick.com"
      },
      "acl": {
        "ipv4-acl-in": "redirect-acl-in"
      }
    }
  ]
}
```

5.6. RBFS Subscriber Filters

5.6.1. RBFS Subscriber Filters Overview

RBFS Subscriber Filters, also referred to as subscriber ACLs, consist of a set of rules defining packet match criteria and actions. There are separate rules for IPv4 and IPv6 downstream (egress to subscriber) and upstream (ingress from subscriber) packets. These rules support various match criteria and actions, some of which are specific to address families or directions. Each rule is assigned a priority, and the decision between multiple matching rules is based on these priorities, where lower values take precedence.

The available actions include accept, drop, or http-redirect where the last one refers to the RBFS HTTP Redirect Service. When the action is drop, matching traffic is silently discarded. The filters are categorized into two primary types, namely l3v4 for IPv4 and l3v6 for IPv6, applicable to either ingress or egress direction.

To apply these filters to subscribers, there are two ways. They can be applied through the access service-profile or directly using the corresponding RADIUS attributes with the second method taking priority.

About the Match Criteria

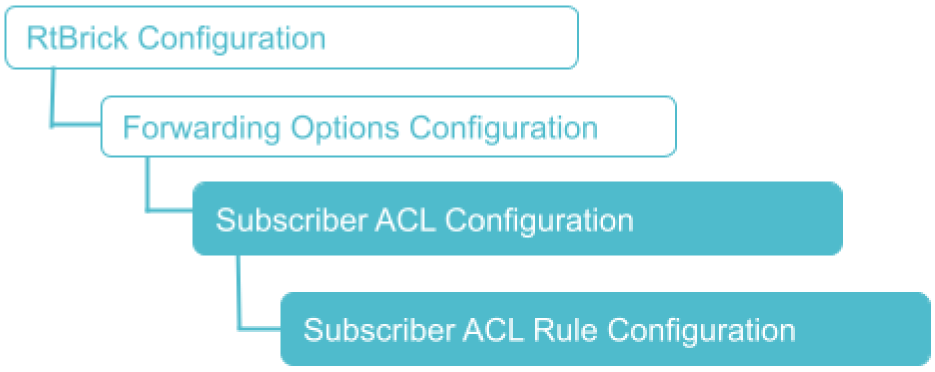
When multiple match criteria are defined within a single rule, they are treated as a logical AND operation, requiring all criteria to be met for the rule to be considered as a match. However, using unsupported match criteria, such as destination-ipv4-subscriber-prefix in ingress (upstream), can potentially lead to session termination. In the case of CoA (Change of Authorization), the filter assignment is rejected using CoA NAK, if such unsupported criteria are encountered.

Even if filters are assigned to a subscriber, those filters are applied globally, indicating that all traffic from all interfaces and subscribers is evaluated against all rules. Consequently, RBFS has introduced specific options to restrict rules to individual subscribers. For ingress (upstream) rules, it is recommended to enable the subscriber-ifl option, ensuring that only traffic received from the corresponding subscriber is matched. With the subscriber-ifl option, packets are matched based on incoming subscriber IFL. However, this option is not supported in egress(downstream), requiring the limitation of traffic using subscriber address prefix information. Thus, RBFS introduced the options source-ipv4-subscriber-prefix, source-ipv6-subscriber-prefix, destination-ipv4-subscriber-prefix, destination-ipv6-subscriber-prefix, source-ipv6-delegated-subscriber-prefix, and destination-ipv6-delegated-subscriber-prefix. With these options enabled, the dynamically assigned subscriber address prefix is automatically integrated into the corresponding filter instance to constrain those rules to a specific subscriber.

Improperly configured filters assigned to one subscriber may create a negative impact on other subscribers as well.

5.6.2. Subscriber Filters Configuration

The configuration hierarchy for Subscriber ACL is illustrated in the diagram.


RtBrick Configuration
Forwarding Options Configuration
Subscriber ACL Configuration
Subscriber ACL Rule Configuration

Configure Subscriber Filters

Syntax:

set forwarding-options subscriber-acl <l3v4|l3v6> **rule** <rule-name> **ordinal** <ordinal-value> <option> <attribute> <value>

Attribute	Description
<l3v4 | l3v6>	Specify l3v4 for IPv4 and l3v6 for IPv6.
<rule-name>	Subscriber ACL rule name.
<ordinal-value>	The mandatory ordinal value is used to differentiate multiple rules within the rule set.
action	The desired action, which can be either permit, drop or http-redirect.
priority <priority-value>	The priority of the rule, where the lower value has a higher priority.
match	The match criteria.

Configuring Subscriber Filter Match Criteria

Syntax:

set forwarding-options subscriber-acl <l3v4|l3v6> **rule** <rule-name> **ordinal** <ordinal-value> **match** <attribute> <value>

Attribute	Description
destination-ipv4-prefix <destination-ipv4-prefix>	Packets are matched when the IPv4 destination address is within the defined prefix.
destination-ipv4-prefix-list <destination-ipv4-prefix-list>	Packets are matched when the IPv4 destination address is within one of the prefixes listed in the defined prefix list.

Attribute	Description
destination-ipv4-subscriber-prefix <true>	Packets are matched when the IPv4 destination address is within the dynamically assigned subscriber IPv4 address prefix (sometimes, referred to framed prefix). Consequently, this option shares similarity to the destination-ipv4-prefix, where the prefix is automatically set to the dynamic subscriber prefix. This option is allowed only in the egress (downstream) direction.
destination-ipv6-prefix <destination-ipv6-prefix>	Packets are matched when the IPv6 destination address is within the defined prefix.
destination-ipv6-prefix-list <destination-ipv6-prefix-list>	Packets are matched when the IPv6 destination address is within one of the prefixes listed in the defined prefix list.
destination-ipv6-subscriber-prefix <true>	Packets are matched when the IPv6 destination address is within the dynamically assigned subscriber IPv6 address prefix (sometimes, referred to framed prefix). Consequently, this option shares similarity to the destination-ipv4-prefix, where the prefix is automatically set to the dynamic subscriber prefix. This option is allowed only in the egress (downstream) direction.
destination-ipv6-delegated-subscriber-prefix <destination-ipv6-delegated-subscriber-prefix>	This option is similar to the destination-ipv6-subscriber-prefix using the dynamically delegated prefix instead.
source-ipv4-prefix <source-ipv4-prefix>	Packets are matched when the IPv4 source address is within the defined prefix.
source-ipv4-prefix-list <source-ipv4-prefix-list>	Packets are matched when the IPv4 source address is within one of the prefixes listed in the defined prefix list.

Attribute	Description
source-ipv4-subscriber-prefix <source-ipv4-subscriber-prefix>	Packets are matched when the IPv4 source address is within the dynamically assigned subscriber IPv4 address prefix (sometimes, referred to framed prefix). Consequently, this option shares similarity to source-ipv4-prefix, where the prefix is automatically set to the dynamic subscriber prefix. This option is allowed only in the ingress (upstream) direction.
source-ipv6-prefix <source-ipv6-prefix>	Packets are matched when the IPv6 source address is within the defined prefix.
source-ipv6-prefix-list <source-ipv6-prefix-list>	Packets are matched when the IPv6 source address is within one of the prefixes listed in the defined prefix list.
source-ipv6-subscriber-prefix <source-ipv6-subscriber-prefix>	Packets are matched when the IPv6 source address is within the dynamically assigned subscriber IPv6 address prefix (sometimes, referred to framed prefix). Consequently, this option shares similarity to source-ipv4-prefix, where the prefix is automatically set to the dynamic subscriber prefix. This option is allowed only in the ingress (upstream) direction only.
source-ipv6-delegated-subscriber-prefix true	This option is similar to the source-ipv6-subscriber-prefix using the dynamically delegated prefix instead. ℹ To disable this configuration, use the delete form of the command.
source-l4-port <source-l4-port>	Packets are matched based on the layer 4 source port (TCP or UDP port).
destination-l4-port <destination-l4-port>	Packets are matched based on the layer 4 destination port (TCP or UDP port).

Attribute	Description
ip-protocol <ip-protocol>	Packets are matched depending on the IP protocol (TCP or UDP). However, this option is not compatible with the MPLS-encapsulated traffic. Consequently, filtering based on IP protocol is not possible for MPLS traffic received from the core. For instance, it is feasible to drop all traffic to port 80, but it is not possible to selectively drop only TCP traffic while permitting UDP traffic when receiving traffic with an MPLS label.
subscriber-ifl <true>	Packets are matched based on incoming subscriber IFL. This option is allowed only in the ingress (upstream) direction.

Configuring Subscriber Filter Actions

Syntax:

set forwarding-options subscriber-acl <l3v4|l3v6> **rule** <rule-name> **ordinal** <ordinal-value> **action** <attribute> <value>

Attribute	Description
permit	Forward packets.
drop	Silently discard packets.
http-redirect true	Specify true to enable the redirect service. To disable http-redirect, use the delete form of the command.

The following example shows an IPv4 subscriber filter configuration in which the applied ACL is ipv4-acl-in.

```
supervisor@rtbrick: cfg> show config forwarding-options subscriber-acl l3v4
{
  "rtbrick-config:l3v4": {
    "rule": [
      {
        "rule-name": "ipv4-acl-in",
        "ordinal": [
          {
            "ordinal-value": 1,
            "match": {
              "destination-l4-port": 80,
              "ip-protocol": "TCP"
            },
            "action": {
              "http-redirect": "true"
            },
            "priority": 1001
          }
        ]
      }
    ]
  }
}
```

Attaching the ACL Rule

Syntax:

set access service-profile profile-name <profile-name> <attribute> <value>

Attribute	Description
<profile-name>	Service profile name.
<http-redirect>	HTTP redirect service configuration.
<url>	HTTP redirect target URL.
<acl>	Subscriber ACL (filter) configuration.
<ipv4-acl-in>	IPv4 upstream ACL (ingress from subscriber).
<ipv4-acl-out>	IPv4 downstream ACL (egress to subscriber).
<ipv6-acl-in>	IPv6 upstream ACL (ingress from subscriber).
<ipv6-acl-out>	IPv6 downstream ACL (egress to subscriber).

5.6.3. Subscriber Filters Operational Commands

Subscriber Filters Show Commands

The show commands provide detailed information about the subscriber filter.

Subscriber ACL (Filter) Information

The show subscriber <id> acl command provides a comprehensive list of all ACL instances initiated for the subscriber, including RBFS Subscriber Filters and Ascend Data Filters (ADF). Additionally, this command provides the option to view detailed information for each filter instance by appending the corresponding filter name. Consequently, the filters are displayed with all variables, such as destination-ipv4-subscriber-prefix, replaced by their actual prefixes for clarity.

Syntax:

show subscriber <subscriber-id> **acl** <acl-name>

Example 1: ACL summary information for a specific subscriber

```
supervisor@rtbrick: op> show subscriber 216454257090494476 acl
Rule                                                 Ordinal     Priority     Type
subscriber-filter-ipoe-0/1/49/216454257090494476     0           1
subscriber_framed_ipv4_filters
                                                     1           2
subscriber_framed_ipv6_filters
                                                     1           2
subscriber_delegated_ipv6_filters
```

Example 2: Summary information for a specific subscriber ACL

```
supervisor@rtbrick: op> show subscriber 1369375761697341441 acl ipv4-acl-in-ipoe-
lag-1/1369375761697341441
Rule: ipv4-acl-in-ipoe-lag-1/1369375761697341441
  ACL type: l3v4
  Ordinal: 1
  Priority: 1001
    Match:
      Direction: ingress
      Destination L4 port: 80
      IP protocol: TCP
    Action:
      HTTP-redirect: True       URL: www.rtbrick.com
```

5.7. Lawful Interception

5.7.1. Lawful Interception Overview

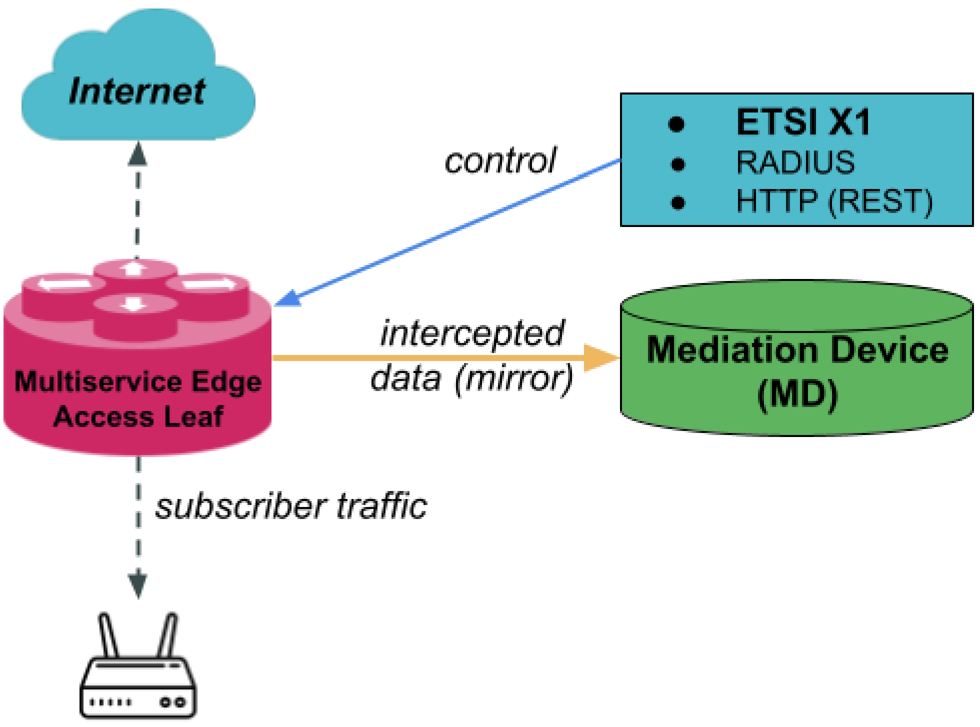
Lawful Interception (LI) is a legal requirement mandated by national regulations in many regions. It enables authorized Law Enforcement Agencies (LEA) to access specific communication data from Communication Service Providers (CSP).

In RBFS, Lawful Interception enables the selective capture of subscriber sessions in both directions (upstream and downstream). The intercepted traffic is tunneled to a Mediation Device (MD), along with metadata indicating the direction of capture and a reference to the intercepted connection or session.

RBFS supports LI in accordance with applicable technical standards and regulatory requirements. The system enables dynamic rule provisioning and role-based access control, ensuring that only authorized personnel can configure or view LI-related information. All interception activities can be logged for auditing and compliance purposes.

Components of Lawful Interception

The figure below shows the different components of the LI solution.



RBFS supports multiple integration methods to implement Lawful Interception (LI), providing flexibility to match the operator’s legal and technical requirements. The following three interfaces are supported for provisioning and managing interception sessions:

1. ETSI X1 Interface (see ETSI TS 101 671)
2. RADIUS Access-Accept and CoA using encrypted attributes
3. HTTP (REST) API

All three methods allow RBFS to set up interception sessions with precise control over subscriber identification, traffic direction, and metadata enrichment. Intercepted traffic is then tunneled to a Mediation Device (MD) using protocols compliant with national regulatory requirements. Role-based access control and full logging ensure that all LI-related actions are secure, traceable, and compliant.

Supported Platforms

Feature availability may vary between hardware platforms. Refer to the Platform Guide for detailed information on which features and sub-features are supported on each specific platform.

5.7.2. ETSI X1

Lawful Interception (LI) involves defined interfaces between Law Enforcement Agencies (LEAs) and Communication Service Providers (CSPs), commonly referred to as handover interfaces. Within the CSP domain, LI defines the following three interfaces:

- **X1** for interception management and administrative control
- **X2** for delivery of intercept-related information (IRI)
- **X3** for delivery of intercepted content (CC)

RBFS implements the X1 interface in compliance with **ETSI TS 101 671**. This interface is used for provisioning, modifying, and terminating interception sessions and forms the control plane between the LI management system and RBFS.

Refer to the “LIX1 Configuration” section below for details on configuring the LIX1 administration interface.

LIX1 Configuration

In this section, you will find the configurations related to the LIX1 interface.

Syntax:

set lawful-intercept protocol <protocol-name> <attribute> <value>

Attribute	Description
<protocol-name>	Enables lawful intercept on the switch. Supported Value: x1.

Attribute	Description
administrative-function-endpoint <administrative-function-endpoint>	Specifies the ADMF endpoint URL for sending requests.
administrative-function-id <administrative-function-id>	Specifies the ADMF identifier.
mediation-device-instance <mediation-device-instance>	Specifies the routing instance that hosts the mediation devices.
mutual-tls	Global mutual TLS configuration.
mutual-tls client authentication <certificate-name>	(Optional) Name of the certificate.
mutual-tls client authentication <certificate-name> certificate <certificate>	(Optional) Specifies the certificate PEM data in base64 encoding. If this value is not specified, it defaults to the server certificate.
mutual-tls client authentication <certificate-name> key-encrypted-text <key-encrypted-text>	(Optional) Specifies the Certificate key in an encrypted format.
mutual-tls client authentication <certificate-name> key-plain-text <key-plain-text>	(Optional) Specifies the Certificate key in base64 encoding. If this value is not specified, it defaults to the server authentication key.
mutual-tls client root-ca <root-ca>	(Optional) Specifies the trusted CA in base64 encoding. This is mandatory for a self-signed certificate.

Attribute	Description
mutual-tls server certificate <certificate>	Specifies the certificate PEM data in base64 encoding.
mutual-tls server client-ca <client-ca>	Specifies the trusted client CAs in base64 encoding.
mutual-tls server key <key-encrypted-text>	Specifies the certificate key in base64 encoding.
network-element-id <network-element-id>	The network element ID of the network element in the ADMF.
network-element-path <network-element-path>	(Optional) The context-path for all incoming protocol requests. Default: /X1/NE.
sync-timeout <5-60>	(Optional) The maximum interval for completing protocol requests synchronously in seconds. Default: 5.
async-timeout <10-120>	(Optional) The maximum interval for completing protocol requests asynchronously in seconds. Default: 15 seconds.
hold-time <60-86400>	(Optional) Specifies the hold time in seconds for the LI tasks if no keepalive messages are seen from ADMF. Default: 3600 seconds. If both sync and async-timeout are configured, async-timeout has to be at least twice the value of sync-timeout.

Invoking LI with CURL

As shown in the example below, you can use CURL to invoke LI.

```
sudo curl --location 'http://localhost/hostconfd/api/v1/li' --header 'Content-
Type: application/json' --data '<Insert data>'  --unix-socket
/var/run/rtbrick/hostconfd_sock/unix.sock -v
```

Enabling Lawful Interception

RBFS restricts access to lawful interception data, making it accessible only to users with the highest privilege level or to explicitly defined LI operator roles.

RBFS supports lawful interception for PPPoE, L2TP, and IPoE subscribers.

5.7.3. RADIUS Lawful Interception

RBFS supports control of Lawful Interception (LI) through RADIUS, using either the Access-Accept or Change of Authorization (CoA) messages. To initiate or manage an LI session via RADIUS, all required LI attributes must be included in the RADIUS Access-Accept or CoA request.

These attributes are salt-encrypted using the algorithm defined in RFC 2868, originally specified for the Tunnel-Password attribute. As this encryption method is defined only for Access-Accept messages, RBFS uses a modified approach for CoA compatibility: the RADIUS Request Authenticator field is replaced with 16 zero bytes. This approach is widely adopted and compatible with multiple RADIUS server implementations.

RFC and draft compliance for RADIUS-based LI is partial unless explicitly stated otherwise.

VSA 26-50058-140 - RtBrick-LI-Action (salt encrypted integer)

Value	Code	Description
NOOP	0	No action / Ignore LI request
ON	1	Start LI / Add LI request
OFF	2	Stop LI / Delete LI request

VSA 26-50058-141 - RtBrick-LI-Identifier (salt encrypted integer) Device unique lawful interception identifier (LIID) within the range from 1 to 4194303.

VSA 26-50058-142 - RtBrick-LI-Direction (salt encrypted integer)

Value	Code	Description
INGRESS	1	Ingress mirroring only (from subscriber)
EGRESS	2	Egress mirroring only (to subscriber)

Value	Code	Description
BOTH	3	Bidirectional mirroring (from and to the subscriber)

VSA 26-50058-143 - RtBrick-LI-MED-Instance (salt encrypted string) The routing instance through which the mediation device is reachable.

VSA 26-50058-144 - RtBrick-LI-MED-IP (salt encrypted IPv4 address) IPv4 address of the mediation device.

VSA 26-50058-145 - RtBrick-LI-MED-Port (salt encrypted integer) UDP port between 49152 and 65535 set in the mirrored traffic

To prevent exposure of LI activity based on the presence of specific attributes, RBFS supports the special LI action NOOP. This action can be used to insert fake or decoy LI requests that have no operational effect but obfuscate detection attempts based on attribute presence alone.

For additional confidentiality, failed LI activations triggered via RADIUS do not return an error or NAK in the RADIUS response. This prevents external systems from distinguishing between valid and NOOP-based requests based on response behavior.

LI requests received via RADIUS, whether real or NOOP, are handled uniformly and appear in the same internal BDS table as requests provisioned through the X1 interface or the HTTP (REST) API.

5.7.4. RBFS Operational State API

The RBFS Operational State API provides endpoints for enabling and disabling LI on a per-subscriber basis:

- A HTTP POST request to `/subscribers/{subscriber_id}/enableLI?id={li_id}&direction={li_direction}&med_ip={med_ip}&med_instance={med_instance}&med_port={med_port}` enables LI for the specified subscriber
- A HTTP POST request to `/subscribers/{subscriber_id}/disableLI?id={li_id}` disable LI for the specified subscriber

The table below lists the request parameters:

Parameter Name	Description
subscriber_id	Subscriber identifier that is generated by RBFS, for example, 72339069014638701.
id	Identifier for Lawful Interception. This is a unique Identifier used by the mediation device to identify the intercepted subscriber. The range can be between 1 to 4194303.
direction	LI direction. Values are: INGRESS, EGRESS, BOTH.
med_instance	VRF instance through which the mediation device is reachable.
med_ip	IPv4 address of the mediation device
med_port	UDP port(MD)(49152-65535), mirrored traffic is forwarded

All parameters are mandatory to enable LI.

Request Examples

Enabling LI:

The example below shows a curl command to enable LI:

```
curl -i -H "Content-Type: application/json" -X POST -d
http://198.51.100.76:19091/api/v1/rbfs/elements/rtbrick/services/opsd/proxy/subscr
ibers/72339069014639042/enableLI?id=66666&direction=BOTH&med_instance=libox&med_ip
=10.0.0.1&med_port=49153
```

Disabling LI:

The example below shows a curl command to disable LI.

```
curl -i -H "Content-Type: application/json" -X POST -d
http://198.51.100.76:19091/api/v1/rbfs/elements/rtbrick/services/opsd/proxy/subscr
ibers/72339069014639042/disableLI?id=66666
```

5.8. L2BSA

5.8.1. L2BSA Overview

Layer 2 Bitstream Access (L2BSA) refers to a scenario in which a service provider makes his access infrastructure available to other service providers. These are retail service providers who provide their Internet services. In Germany, this service is mandated by the Federal Network Agency (German: Bundesnetzagentur or BNetzA) which is the regulatory office for electricity, gas, telecommunications, post, and railway markets. It is a federal agency of the Federal Ministry for Economic Affairs and Energy and is headquartered in Bonn, Germany.

The definition of the L2BSA service was defined by the so-called NGA Forum, an advisory board founded in May 2010 by the Bundesnetzagentur for promoting dialogue between the Bundesnetzagentur, network operators, manufacturers, states and local authorities on NGA rollout.

https://www.bundesnetzagentur.de/DE/Fachthemen/Telekommunikation/Breitband/NGA-Forum/start.html

The L2BSA specification defines two interfaces, the so-called U interface (User Interface) at the customer location and the A10-NSP interface (A10 Network Service Provider) between the service provider networks. Those interface types were originally introduced in the Broadband Forum TR-101 (Migration to Ethernet-based Broadband Aggregation).

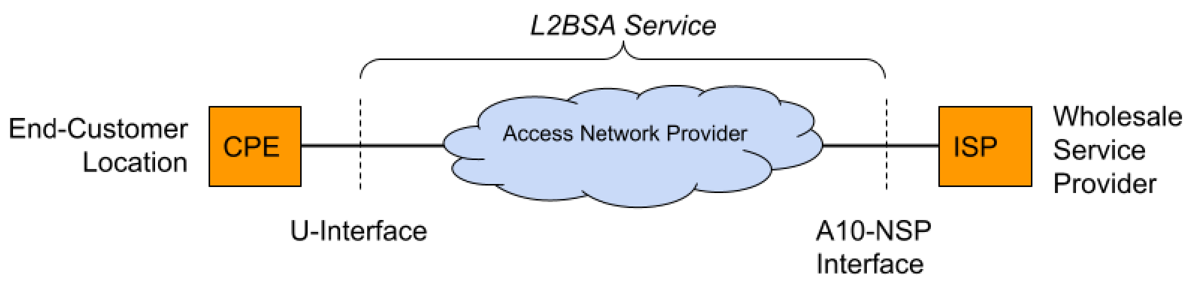


Figure 21. L2BSA Interfaces

The U interface is defined as a transparent Layer 2 interface which can be used with or without VLAN tags by the wholesale service providers. This means that some CPE will send their traffic untagged while another CPE is configured for tagged traffic which needs to be transparently forwarded between the U interface and the A10-NSP interface.

The A10-NSP interface is defined as a link aggregation bundle interface with one or more interfaces and LACP enabled or disabled. All traffic on this interface is at least single-tagged with the so-called S-VLAN tag which identifies the U interface. This limits the amount of L2BSA services to 4094 per A10-NSP interface caused by the usable VLAN range. Therefore some providers need multiple A10-NSP interfaces if they need to address more than the 4094 services.

The term A10 relates to the end-to-end ADSL network reference model depicted in the figure below. The Core Network reference model is a subset of the end-to-end architecture; it is composed of two functional blocks and three reference points. The Access Node and the Regional Broadband Network are the two functional blocks. U, V and A10 are the three reference points.

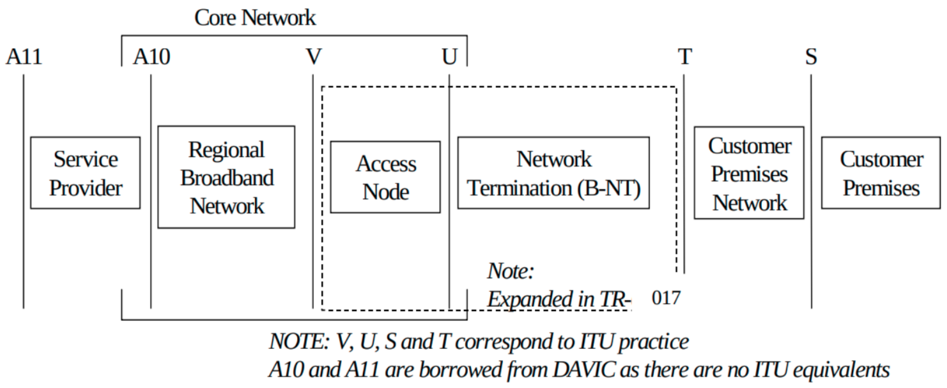

Core Network
A11
A10
V
U
T
S
Service Provider
Regional Broadband Network
Access Node
Network Termination (B-NT)
Customer Premises Network
Customer Premises
Note: Expanded in TR-017
NOTE: V, U, S and T correspond to ITU practice
A10 and A11 are borrowed from DAVIC as there are no ITU equivalents

Figure 22. TR-025

Source: https://www.broadband-forum.org/download/TR-025.pdf

The mapping between the U interface and A10-NSP/S-VLAN is managed by the L2BSA service provider and typically changes frequently triggered by re-provisioning actions. Therefore all PPPoE discovery, as well as DHCPv4/v6 packets, must be enriched with additional line identification headers (Agent-Remote-Id, Agent-Circuit-Id, Actual-Data-Rate, …) by the L2BSA service provider in the upstream direction (from U to A10-NSP interface) to allow the wholesale provider to identify the actual U interface for traffic received on the A10-NSP interface. This functionality is referred to as the intermediate agent functionality.

The U interface is terminated on the access node which might be an OLT for fiber-based access or MSAN for xDSL, therefore the U interface can also be called the customer access line or port. The access node will add additional VLAN for each access line to uniquely identify them between the access node and leaf switch. This VLAN is called the ANP tag (Access-Node-Port).

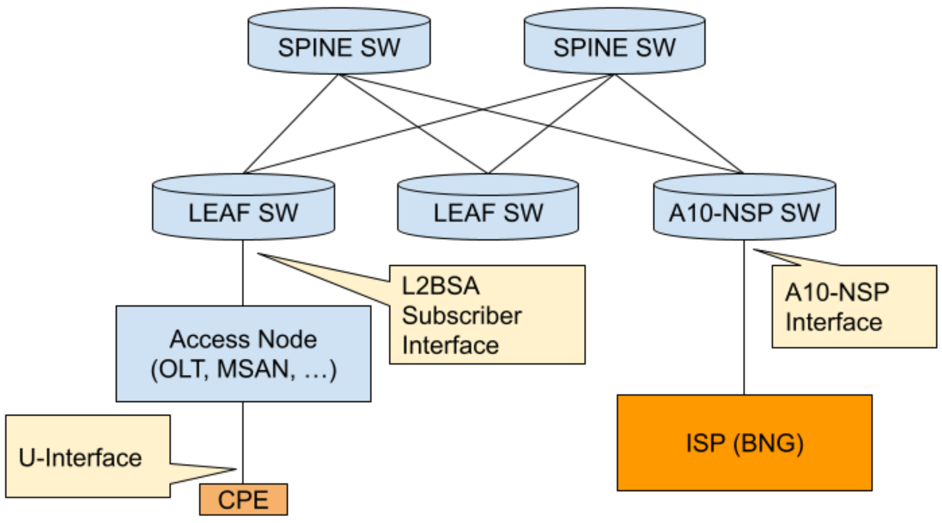


Figure 23. L2BSA Fabric

For L2BSA services all traffic received with a given ANP tag will be transparently forwarded to the corresponding A10-NSP interface via Layer 2 cross-connect (L2X) services. We introduce a new subscriber type for L2BSA in our RBFS access infrastructure to manage those services on the leaf switches and to allow further advanced access services like QoS or subscriber accounting.

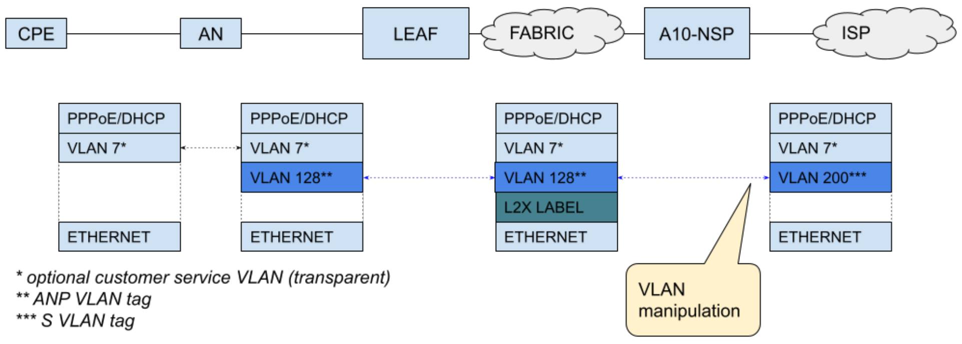


Figure 24. L2BSA Traffic

L2BSA Services and Subscribers

RBFS distinguishes between L2BSA services and subscribers. L2BSA services are created using the Operational State API (/l2bsa) stored as ephemeral objects in the configuration daemon. Those objects are excluded from the actual configuration and not restored after reboot. The L2BSA service represents a defined interface to add, update and delete L2BSA services. Those services are managed by external orchestrators or service managers and are not changed or deleted by RBFS.

Adding such an L2BSA service triggers the creation of a corresponding L2BSA subscriber which represents the internal state. Each L2BSA subscriber is a full-access subscriber and therefore most subscriber features will work for L2BSA as well. This includes features like RADIUS authentication, accounting, CoA or dynamic QoS. Each L2BSA subscriber persists as long as the service is still present and also if the subscriber has already been terminated. This allows external applications to collect all states and counters before the subscriber is finally deleted triggered by L2BSA service delete.

A10-NSP L2X Services

For each L2BSA service, there must also be an A10-NSP L2X endpoint created using the Operational State API (/a10nsp/l2x).

The A10-NSP endpoint is the place where all VLAN manipulation is done meaning where ANP and S-VLAN are swapped.

Supported Platforms

Support for the following underlays are available:

- IS-IS Segment Routing
- BGP Segment Routing

Not all features are necessarily supported on each hardware platform. Refer to the Platform Guide for the features and the sub-features that are or are not supported by each platform.

5.8.2. L2BSA Configuration

This document assumes a working base configuration explaining the L2BSA specific additions only.

AAA Profile Configuration

Each L2BSA service refers to a mandatory AAA configuration profile as explained in detail in the Subscriber Management Configuration Guide. The following example shows the minimum configuration with authentication and accounting disabled.

```
supervisor@leaf1: cfg> show config access aaa-profile aaa-l2bsa-default
{
    "rtbrick-config:aaa-profile": [
      {
        "profile-name": "aaa-l2bsa-default",
        "authentication": {
          "order": "NONE"
        },
        "accounting": {
          "order": "NONE"
        }
      }
    ]
  }
```

Intermediate Agent Configuration

DEPRECATED!

The functionality of the intermediate agent is designed to be implemented within the access node, which could be an OLT for fiber-based access or MSAN for xDSL. RBFS is technically prepared to fulfill the role of an intermediate agent in exceptional circumstances where the access node does not offer this function.

Please note that the intermediate agent function is deprecatd and will be removed in future releases!

The RBFS intermediate agent is capable of appending headers in the upstream direction but cannot strip them again in the downstream. Consequently, it is compatible only with PPPoE and DHCPv4. For DHCPv4, it should be noted that headers added upstream are not removed downstream if echoed back by the DHCP server, which is the usual behavior.

To enable the intermediate agent function, it must be enabled for each physical interface used by L2BSA services.

```
supervisor@leaf1: cfg> set access l2bsa intermediate-agent interface ifp-0/1/23
  <cr>
  pppoe-enable          Enable/disable the L2BSA intermediate agent for PPPoE
Discovery
  dhcpv4-enable         Enable/disable the L2BSA intermediate agent for DHCPv4
  vlan-mode             L2BSA intermediate agent VLAN mode
  inner-vlan-min        Inner VLAN min
  inner-vlan-max        Inner VLAN max
```

Attribute	Description
pppoe-enable	This option allows enabling/disabling the intermediate agent function for PPPoE discovery packets. **Default:** false (disabled)
dhcpv4-enable	This option allows enabling/disabling the intermediate agent function for DHCPv4 packets. **Default:** false (disabled)
vlan-mode	This option enables the intermediate agent function for single, double, or single and double-tagged packets. **Default:** SINGLE_DOUBLE **Values:** SINGLE, DOUBLE, SINGLE_DOUBLE
inner-vlan-min/max	This option is applicable for double-tagged packets only and allows limiting the intermediate agent function to a given inner VLAN range. **Default:** all

The following common example enables the intermediate agent for all single-tagged PPPoE/DHCPv4 packets and double-tagged PPPoE/DHCPv4 packets with inner VLAN between 1 and 3000.

```
supervisor@leaf1: cfg> show config access l2bsa intermediate-agent
{
    "rtbrick-config:intermediate-agent": {
      "interface": [
        {
          "interface-name": "ifp-0/1/23",
          "inner-vlan-min": 1,
          "inner-vlan-max": 3000,
          "pppoe-enable": "true",
          "dhcpv4-enable": "true"
        }
      ]
    }
  }
```

5.8.3. L2BSA Operational Commands

The following commands allow to list or clear all L2BSA services from CLI.

```
supervisor@leaf1: op> show l2bsa service
Interface        ANP VLAN     Timestamp
ifp-0/0/1     1000         Mon Aug 30 09:21:14 GMT +0000 2021

supervisor@leaf1: op> clear l2bsa service all
Success
```

L2BSA subscribers can be managed like any other subscriber using the subscriber CLI commands.

```
supervisor@leaf1: op> show subscriber
Subscriber-Id          Interface        VLAN      Type   State
281479271677954        ifp-0/1/23     1000:0    L2BSA  ESTABLISHED

supervisor@leaf1: op> show subscriber filter type L2BSA
Subscriber-Id          Interface        VLAN      Type   State
281479271677954        ifp-0/1/23     1000:0    L2BSA  ESTABLISHED

supervisor@leaf1: op> show subscriber 281479271677954 detail
Subscriber-Id: 281479271677954
    Type: L2BSA
    State: ESTABLISHED
    Created: Mon Aug 30 09:23:34 GMT +0000 2021
    Interface: ifp-0/1/23
    Outer VLAN: 1000
    IFL: l2bsa-0/0/1/281479271677954
    Username: ifp-0/1/23:1000@l2bsa
    Agent-Remote-Id: DEU.L2BSA.API01
    Agent-Circuit-Id: 0.0.0.0/0.0.0.0 eth 01
    AAA-Profile: aaa-l2bsa-default
    Session-Timeout: 0 (disabled)
    Idle-Timeout: 0 (disabled)
    L2X:
        Instance: default
        Nexthop: 198.51.100.101 (ipv4/labeled-unicast)
        Ingress Label (TX): 1337
        Egress Label (RX): 7331
    Accounting:
        Session-Id: 281479271677954:1630315414
        Start-Time: 2021-08-30T09:23:34.847166+0000
        Interims Interval: 0 seconds
```

The CLI also provides some global intermediate agent function packet statistics about traffic received in the control plane which belong to L2BSA subscribers.

```
supervisor@leaf1: op> show l2bsa intermediate-agent statistics
PPPoE Discovery Sent        : 0
PPPoE Discovery Updated     : 0
PPPoE Discovery Dropped     : 0
```

The sent statistics count all intermediate agent traffic forwarded back to L2X while updated only counts those which are also updated or enriched by access line information. Dropped are those packets received but dropped by the L2BSA subscriber state which might be the case if the L2BSA subscriber is still present but in a terminating state.

5.8.4. L2BSA Operational State API

The API is split into the L2BSA and A10-NSP L2X service API. Details about those API endpoints, schema and attributes can be found in the official RBFS Operational State OpenAPI documentation.

L2BSA Services API

Each L2BSA service is uniquely identified by physical interface name (IFP) and ANP VLAN identifier. All other attributes of L2BSA services can be changed dynamically by replacing the existing service object by doing a full update meaning that updates must include all attributes and RBFS will automatically recognize the changes to the last version to apply those incrementally.

Endpoint	Description
GET /l2bsa	This endpoint returns a list of all L2BSA services.
PUT /l2bsa	This endpoint replaces all L2BSA services on the system by adding new ones, updating existing and deleting those not included in the request but present on the system. Therefore this endpoint can be used to delete all services if an an empty list is provided in the request body.
GET /l2bsa/{ifp_name}	This endpoint returns a list of all L2BSA services on the given physical interface (IFP).
PUT /l2bsa/{ifp_name}	This endpoint works similarly to PUT /l2bsa but affects only services matching the given physical interface (IFP). Therefore this endpoint can be used to delete all services of a given physical interface if an empty list is provided in the request body.
GET /l2bsa/{ifp_name}/{anp}	This endpoint returns a single L2BSA service or 404 if no matching service is found.

Endpoint	Description
PUT /l2bsa/{ifp_name}/{anp}	This endpoint adds or updates a single L2BSA service.
DELETE /l2bsa/{ifp_name}/{anp}	This endpoint deletes a single L2BSA service.

The following example shows an L2BSA service request body with only the mandatory attributes set.

```
{
    "ifp_name": "ifp-0/0/1",
    "anp_vlan": 128,
    "aaa_profile_name": "aaa-l2bsa-default",
    "l2x": {
      "ingress_nexthop": "2001:db8:0:30::",
      "ingress_service_label": 1001,
      "egress_service_label": 1002
    }
}
```

The L2X ingress next-hop is typically the fabric loopback IPv4 or IPv6 address of the corresponding switch where the corresponding A10-NSP L2X service is placed. The ingress label is the upstream service label added for traffic sent to the corresponding next-hop (sent label). The egress label is the downstream service label where traffic sent from the next-hop is expected to be received (receive label). This label must be unique per switch.

Only labels greater than 100.000 should be used for the L2X service labels because labels between 16 - 100.000 are reserved by other applications. L2X service labels must not conflict with other L2X service labels and the same label can be used for downstream and upstream.

L2BSA supports also dynamic QoS which can be controlled by RADIUS if enabled or directly via API as shown in the following example.

```
{
    "ifp_name": "ifp-0/0/1",
    "anp_vlan": 128,
    "aaa_profile_name": "aaa-l2bsa-default",
    "l2x": {
      "ingress_nexthop": "2001:db8:0:30::",
      "ingress_service_label": 1001,
      "egress_service_label": 1002
    },
    "qos": {
      "qos_profile_name": "l2bsa-qos-default",
      "parent_scheduler": "pon1",
      "shaper":
"name=shaper_session,high=14000,low=2000;name=shaper_voice,high=2000",
      "policer": "level=4,cir=1m,cbs=256;level=3,cir=2m,cbs=512",
      "queue": "name=BestEffort,size=65000;name=Voice,size=20000"
    }
}
```

The QoS configuration profile is used to instantiate the QoS resources for the subscriber including schedulers, queues and shapers. This profile can be set using the qos_profile_name attribute. All dynamic QoS settings like MFC, queue sizes, shaper and policer rates will be reset if this attribute is present also if not changed. Additional shaper or policer settings included will be applied to the new QoS configuration after reset.

The assigned shaper instances can be updated using the shaper attribute which will apply to the QoS instance of the corresponding subscriber-only, but not to the other subscribers using the same QoS profile. It is only possible to update existing shapers dynamically but it is not possible to create a new shaper.

```
"shaper": "<shaper-name>,<high-kbps>,<low-kbps>;<shaper-name>,…"
```

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
"shaper": "name=<shaper-name>,high=<high-kbps>,low=<low-kbps>;…"
```

The policer attribute is also a string that contains a list of multiple policer-level settings separated by a semicolon. Each set contains a level, cir, cbs, pir, pbs, max-cir and max-pir separated by a comma.

```
"policer": "<level>,<cir>,<cbs>,<pir>,<pbs>,<max-cir>,<max-pir>;<level>…"

Example:
"policer": "1,2000,200;2,8000,800;3,0,800;4,0,800"

level: 1 (highest priority) to 4 (lowest priority)
cir: ingress policer committed information rate (kbps)
cbs: ingress policer committed burst size (kbps)
pir: ingress policer peak information rate (kbps)
pbs: ingress policer peak burst size (kbps)
max-cir: max ingress policer committed information rate (kbps)
max-pir: max ingress policer peak information rate (kbps)
```

If PIR and PBS are not defined, the values from CIR and CBS are used as PIR and PBS as well. The max CIR and max PIR attributes are optional default set to unlimited.

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
"policer": "level=<level>,cir=<cir>,cbs=<cbs>,pir=<pir>,pbs=<pbs>,max-cir=<max-
cir>,max-pir=<max-pir>;…"

Example:
"policer": "level=4,cir=1m,cbs=256;cir=2m,cbs=512,level=3"
```

The queue attribute is a string that contains a list of multiple queue settings separated by a semicolon. Each queue setting contains a queue name and queue size in bytes separated by a comma.

```
"queue": "<queue-name>,<size-bytes>;<queue-name>,<size-bytes>;…"
```

This attribute can be also used as a key-value list which is automatically recognized by RBFS.

```
"queue": "name=<queue-name>,size=<size-bytes>;name=…"
```

The subscriber management infrastructure does not check if a referenced QoS profile, shaper, queue or policer exists or not. This is handled by forwarding infrastructure which continues processing the subscriber QoS settings as soon as the missing resource was added.

The parent scheduler element of the scheduler map assigned to the subscriber can be selected with the parent_scheduler attribute. If not present, the scheduler map will be directly bound to the local IFP where the session is established. The parent scheduler can be only set at the beginning and must not be changed.

Providing a QoS parent scheduler that is not present on the corresponding IFP will lead to the black howling of all egress data traffic.

The optional access_line_info attribute contains the access line identification and characteristics attributes used by the intermediate agent function. A detailed list of all supported line attributes and values can be found in the official RBFS Operational State OpenAPI documentation.

For access line attributes there is a difference between attributes not present or set to zero. Those not present will be also not added by the intermediate agent function whereas those set to zero will be added by the intermediate agent function with value also set to zero.

Example:

```
/usr/bin/curl --location --request PUT
'http://198.51.100.35:19091/api/v1/rbfs/elements/leaf1/services/opsd/proxy/l2bsa/i
fp-0%2f0%2f1/128' --header 'Content-Type: application/json' --data-raw '{
    "ifp_name": "ifp-0/0/1",
    "anp_vlan": 128,
    "aaa_profile_name": "aaa-l2bsa-default",
    "l2x": {
        "ingress_nexthop": "2001:db8:0:30::",
        "ingress_service_label": 1001,
        "egress_service_label": 1002
    },
    "qos": {
        "qos_profile_name": "l2bsa-qos-default",
        "parent_scheduler": "pon1",
        "shaper":
"name=shaper_session,high=14000,low=2000;name=shaper_voice,high=2000",
        "policer": "level=4,cir=1m,cbs=128;level=2,cir=1m,cbs=128",
        "queue": "name=BestEffort,size=65000;name=Voice,size=20000"
    },
    "access_line_info": {
        "agent_circuit_id": "0.0.0.0/0.0.0.0 eth 0/0",
        "agent_remote_id": "DEU.RTBRICK.L2BSA01",
        "upstream": {
            "actual_rate": 2048
        },
        "downstream": {
            "actual_rate": 16235
        },
        "pon_type": "GPON"
    }
}'
```

L2BSA Subscribers API

The existing subscriber API was extended to do some common actions using the L2BSA physical interface name (IFP) and ANP VLAN identifier instead of the subscriber-id.

Endpoint	Description
GET /subscribers/l2bsa/{ifp_name}/{anp}	This endpoint is an alias for GET /subscribers/{subscriber_id} returning detailed subscriber information or 404 if no matching subscriber is found.
DELETE /subscribers/l2bsa/{ifp_name}/{anp}	This endpoint is an alias for DELETE /subscribers/{subscriber_id} which terminates a matching L2BSA subscriber.
GET /subscribers/l2bsa/{ifp_name}/{anp}/adjusted-accounting	This endpoint is an alias for GET /subscribers/{subscriber_id}/adjusted-accounting returning the adjusted accounting information or 404 if no matching subscriber is found.

A10-NSP L2X Services API

Endpoint	Description
GET /a10nsp/l2x	This endpoint returns a list of all A10-NSP L2X services.
PUT /a10nsp/l2x	This endpoint replaces all A10-NSP L2X services on the system by adding new ones, updating existing and deleting those not included in the request but present on the system. Therefore this endpoint can be used to delete all services if an an empty list is provided in the request body.
GET /a10nsp/l2x/{lag_interface_name}	This endpoint returns a list of all A10-NSP L2X services on the given the LAG interface.
PUT /a10nsp/l2x/{lag_interface_name}	This endpoint works similarly to PUT /a10nsp/l2x but affects only services matching the given LAG interface. Therefore this endpoint can be used to delete all services of a given the LAG interface if an empty list is provided in the request body.
GET /a10nsp/l2x/{lag_interface_name}/{s_anp}	This endpoint returns a single A10-NSP L2X service or 404 if no matching service is found.

Endpoint	Description
PUT /a10nsp/l2x/{lag_interface_name}/{s_anp}	This endpoint adds or updates a single A10-NSP L2X service.
DELETE /a10nsp/l2x/{lag_interface_name}/{s_anp}	This endpoint deletes a single A10-NSP L2X service.

The following example shows an A10-NSP L2X service request body with only the mandatory attributes set.

```
{
    "lag_interface_name": "lag-1",
    "s_vlan": 100,
    "anp_vlan": 128,
    "l2x": {
        "ingress_nexthop": "2001:db8:0:40::",
        "ingress_service_label": 1002,
        "egress_service_label": 1001
    }
}
```

The L2X ingress next-hop is typically the fabric loopback IPv4 or IPv6 address of the corresponding switch where the corresponding L2BSA service is placed. The ingress label is the upstream service label added for traffic sent to the corresponding next-hop (send label). The egress label is the downstream service label where traffic sent from the next-hop is expected to be received (receive label). This label must be unique per switch.

Only labels greater than 100000 should be used for the L2BSA service labels because labels between 16 - 100000 are reserved by other services. L2BSA service labels must not conflict with other L2X service labels and the same label can be used for downstream and upstream.

Example:

```
/usr/bin/curl --location --request PUT
'http://198.51.100.35:19091/api/v1/rbfs/elements/a10nsp-
sw1/services/opsd/proxy/a10nsp/l2x/lag-1/100' \
--header 'Content-Type: application/json' \
--data-raw '{
    "lag_interface_name": "lag-1",
    "s_vlan": 100,
    "anp_vlan": 128,
    "l2x": {
        "ingress_nexthop": "2001:db8:0:40::",
        "ingress_service_label": 1002,
        "egress_service_label": 1001
    }
}'
```

- L2BSA supports both PPPoE and IPoE subscribers.
- L2BSA supports DHCPv4 and DHCPv6 protocols.

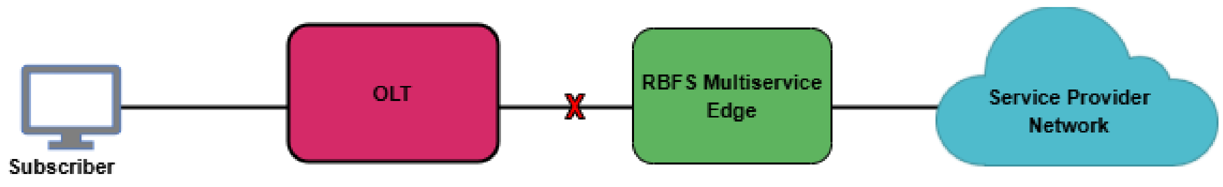
5.9. Redundancy

5.9.1. IPoE Stateful Redundancy for Subscriber Groups

Overview

Node outages and link failures that may occur on an access network can bring down the subscriber services. These network outages affect critical workloads and continuity of business. So, it is essential to set up a network that is resilient and responds quickly to the events and protects the network from outages.

The following diagram represents a simple access network without redundancy.



IPoE Stateful Redundancy protects subscriber services from node or link outages. It provides mechanisms to enhance network resiliency that enables subscriber workloads to remain functional by ensuring a reliable switchover in the event of a node or link outage. With the IPoE Stateful Redundancy, if one node goes down due to node or link failure, another node can automatically take over the services.

- RBFS currently supports stateful redundancy only for IPoE subscribers.
- Redundancy and IPoE relay mode are mutually exclusive and cannot function together on the same device.

- The term 'Multiservice Edge' used in this document refers to a Multiservice Edge platform. The Multiservice Edge platform integrates all BNG functionalities within a single platform. In contrast, the spine-leaf topology distributes these functionalities between spine and leaf platforms based on their respective roles.
- IPoE Stateful Redundancy is supported only on Multiservice Edge platforms. IPoE Stateful Redundancy is not supported on spine-leaf network topology as the spine-leaf topology architecture itself innately provides a redundant and resilient network.

Understanding IPoE Stateful Redundancy

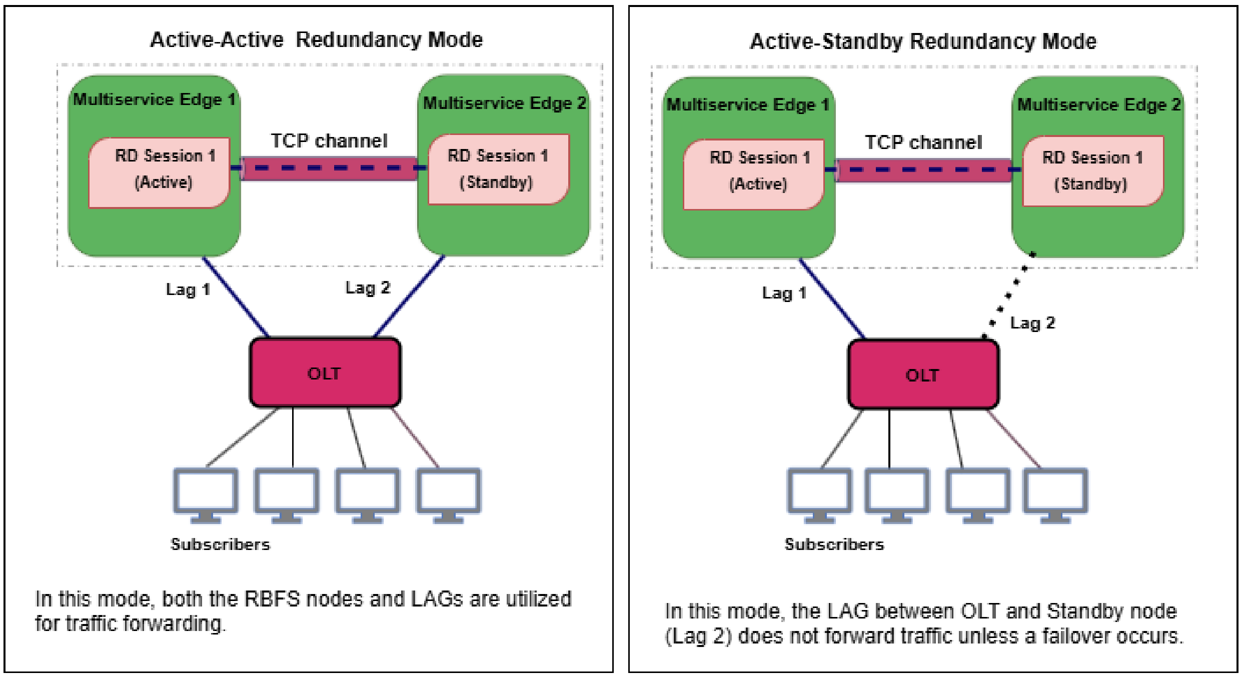
RBFS supports stateful redundancy and fault tolerance through two primary mechanisms:

- Active-Active Redundancy
- Active-Standby Redundancy

You can select the desired redundancy mechanism and configure the system accordingly to enable the feature. By default, the Active-Active (Standby-Forwarding) mode is enabled. If you prefer to use the Active-Standby Redundancy mode, select the Standby-Non-Forwarding option in the Platform Profile configuration. This is a one time configuration unless you choose to switch to the other Redundancy mode.

IPoE Stateful Redundancy is centered around subscriber groups, known as redundancy sessions. IPoE Stateful Redundancy protects subscriber groups using a two-node cluster model, either active-active or active-standby.

The following image illustrates two different redundancy mechanisms: Active-Active and Active-Standby. In Active-Active mode, both BNG nodes and LAGs forward traffic. In Active-Standby mode, only the active node and the LAG (between the OLT and the active node) handle traffic, while the standby node remains idle until a failover occurs.


Active-Active Redundancy Mode
Multiservice Edge 1
Multiservice Edge 2
TCP channel
RD Session 1 (Active)
RD Session 1 (Standby)
Lag 1
Lag 2
OLT
Subscribers
In this mode, both the RBFS nodes and LAGs are utilized for traffic forwarding.
Active-Standby Redundancy Mode
Multiservice Edge 1
Multiservice Edge 2
TCP channel
RD Session 1 (Active)
RD Session 1 (Standby)
Lag 1
Lag 2
OLT
Subscribers
In this mode, the LAG between OLT and Standby node (Lag 2) does not forward traffic unless a failover occurs.

From the perspective of the OLT, there is only one LAG that is distributed across multiple BNGs. This means that the OLT sees one logical link, even though it is physically connected to two separate BNGs. But, in reality, this LAG spans two BNGs, indicating that the OLT is actually connected to two different devices.

Redundancy Session

Redundancy session is a replication mechanism that is used to pair Multiservice Edge nodes for redundancy. Redundancy allows grouping of subscribers, under a redundancy session and each redundancy session is represented by a redundancy session ID. Simply put, an IPoE Stateful Redundancy session represents a redundancy group of subscribers. A redundancy session enables linking the LAG with that particular redundancy session (subscriber group). When you define a value for the redundancy session ID, this ID should be unique and the same for both redundancy pairs. When two nodes get the same session ID, they recognize each other as the peer nodes for a particular redundancy session (subscriber group). The TCP session establishment between the nodes occurs after the pairing with the redundancy session ID. Once the TCP session is established, the nodes use this channel for subscriber data mirroring and synchronization and for health status monitoring.

A Multiservice Edge chassis can contain multiple redundancy sessions. Multiple Multiservice Edge nodes can be active nodes for one or more subscriber groups (redundancy sessions) that serve the subscribers and they can, at the same time, be standby nodes for other subscriber groups.

One node, which is paired for redundancy, can perform subscriber services for more than one redundancy session (subscriber group). The peer node, which is identical to the first node, contains the same subscriber group (redundancy session) as a standby. And in the event that first node goes down due to any outage, the standby node can take over subscriber service for this redundancy session.

IPoE Stateful Redundancy allows running a redundancy session actively on one node and backing up the same redundancy session (subscriber group) on a different (standby) Multiservice Edge node. In IPoE Stateful redundancy, in fact, there is no active node or standby node. It is either an active subscriber group or standby subscriber group. A Multiservice Edge node can be active for a subscriber group and at the same time it can be a standby for a different subscriber group. You can configure a maximum number of 64 redundancy sessions (either active or standby) on a Multiservice Edge node.

Active-Active Redundancy

In the Active-Active Redundancy mode, both of the nodes and their associated LAGs are actively used for traffic forwarding. Both the nodes synchronize their subscriber state information in real-time. This ensures that all the LAG links towards the OLT are utilized for maximum resource efficiency.

If one node fails, the other node can immediately take over the responsibilities with minimal disruption, as it is already synchronized with the active node. This mechanism provides fast failover with minimal traffic disruption and ensures traffic flow.

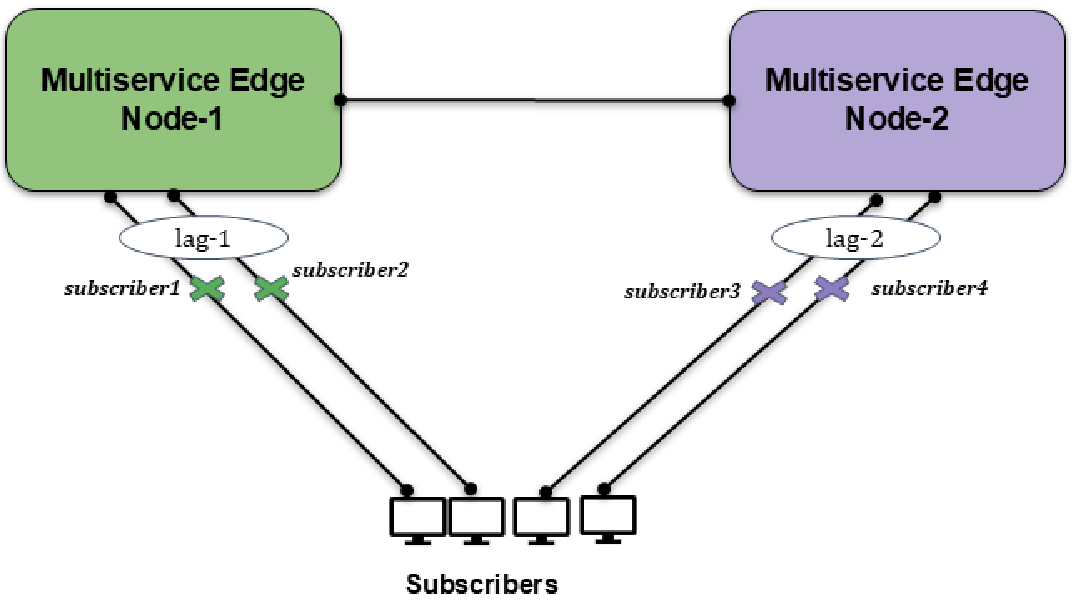
Pinning or Assigning a Subscriber to a Link of a LAG

RBFS uses a pinning mechanism where the system assigns or 'pins' each subscriber to each of the LAG member links between active/standby nodes and the OLT.

Subscribers are assigned during session bringup to member links belonging to the access LAG on both of the nodes. They will be assigned either as locally pinned or remotely pinned. This ensures an equal distribution of subscribers across all member links.

Locally pinned: Refers to subscribers pinned to a member links of the LAG of the Active node.

Remotely pined: Refers to subscribers pinned to a member links of the LAG of the Standby node.



OLT is excluded from the diagram to simplify it and is not directly relevant to what is depicted.

The diagram illustrates the pinning mechanism, where Subscriber 1 is pinned to Link 1 of LAG 1, and Subscriber 2 is pinned to Link 2 of LAG 1. Similarly, Subscriber 3 is pinned to Link 1 of LAG 2, while Subscriber 4 is pinned to Link 2 of LAG 2.

When a member link of the access LAG fails on either the active or standby device in the Active-Active Redundancy mode, the system retriggers the subscriber pinning process to maintain an equal distribution of subscribers across the access LAG member links.

- Locally pinned subscribers will be in 'established' state on the active node and in the standby state on the Standby node.
- Remotely pinned subscribers will be in the 'established non-forwarding' state on the active node and in the 'standby-forwarding' state on the standby node.

Subscriber Pinned to Active Node

The following table provides the various subscriber state for the Active-Active Redundancy.

Subscriber Pinning	State on Active Node	State on Standby Node
Pinned to Active Node	Established	Standby
Pinned to Standby Node	Established Non-Forwarding	Standby-Forwarding

The following example shows the IPoE subscribers pinned to different LAGs and their states.

```
supervisor@rtbrick.net: cfg> show subscriber
Subscriber-Id            Interface         VLAN       Type    State
1369375761697341460      lag-1             1001:1001  IPoE    ESTABLISHED
1369375761697341463      lag-1             1001:1004  IPoE    ESTABLISHED_NO_FWD
2522297266304188429      lag-11            2001:1001  IPoE    STANDBY
2522297266304188430      lag-11            2001:1002  IPoE    STANDBY_FWD
```

The subscriber ID 1369375761697341460 is pinned to 'lag-1' and is in established state on Active node. The subscriber ID 1369375761697341463 is pinned to 'lag-1' is in a established-non-forwarding mode on the active node. The subscriber ID 2522297266304188429 is pinned to 'lag-11' and is in standby mode on the Standby node. The subscriber ID 2522297266304188430 is pinned to 'lag-11' and is in standby-forwarding mode on the Standby node.

Active-Standby Redundancy

In the active-standby node cluster, the active node (for a redundancy session) performs subscriber services. The standby device mirrors concurrent subscriber state data from the active peer (for that redundancy session). Both the nodes, paired for redundancy, keep sending 'keepalive' messages to each other to check the health status.

In this mode, the standby RBFS node does not forward traffic while the active RBFS node is functioning. The standby node stays synchronized with the active node by maintaining up-to-date routing tables, configurations, and session data, but it does not perform any traffic forwarding. If the active node fails, the standby node will take over its role and start forwarding traffic.

Active node handles all subscriber traffic and control plane processing. In this model, the standby LAG is not utilized for traffic unless a failover occurs.

IP Address Pool Management for Redundancy

There is no synchronization between local IP address pools, so an IP address on one BNG will not be reflected in the other BNG’s IP address pool.

So, it is highly recommended to use RADIUS for IP address management. If local pools are used, they must be configured with disjoint address ranges across all nodes. While an IP address assigned by a pool on the active BNG can be transferred to the standby BNG during failover, the local pools are not synchronized.

Subscriber Services on Redundancy

Subscriber services refer to the various features that can be applied to subscribers, such as access control lists, HTTP redirect, lawful interception, CGNAT, IGMP, and Accounting. When deploying these services in a redundancy setup, special handling is required. The behavior of each service varies depending on whether the Redundancy mode is active-active or active-standby.

The following table provides information about the various services supported on Active-Active Redundancy and Active-Standby Redundancy.

Service	Active-Active	Active-Standby
Accounting	Supported: Both BNGs perform accounting and RADIUS aggregates counters.	Supported: Only the active BNG performs accounting.
Lawful Interception	Supported: LI must be enabled on both BNGs.	Supported: LI happens in active BNG and the data is synced to standby BNG.
HTTP Redirect	Supported: Enabled on both BNGs.	Supported: Enabled only in active BNG and synced to standby BNG.
CGNAT	Unsupported	Unsupported
Subscriber ACL	Supported: ACLs are programmed on both BNGs.	Supported: ACLs programmed only in active BNG and synced to standby BNG.

Service	Active-Active	Active-Standby
ADF (Application Detection and Filtering)	Unsupported	Unsupported
Dynamic QoS Profiles	Supported: QoS profile changes via CoA sync on both BNGs.	Supported: QoS profile changes via CoA sync to standby BNG.
Dynamic QoS Parameters	Unsupported	Unsupported
IGMP (Multicast Services)	Unsupported	Unsupported

Inter-BNG Connectivity

Multiservice Edge platforms, paired for redundancy, are connected with a Redundancy (RD) TCP session. To maintain state consistency, a TCP-based Redundancy session is established between the active and standby BNGs. RBFS manages redundancy sessions over IP/MPLS connectivity between the two BNGs.

This RD TCP connection can be formed either directly or through the core network. It may use any routing protocol (for unicast reachability) and LDP or MPLS-SR derivates(for labeled unicast reachability) between the active and standby nodes. The session between the Multiservice Edge pairs establishes connectivity and the link is used to send 'keepalive' messages and data mirroring for subscriber state synchronization.

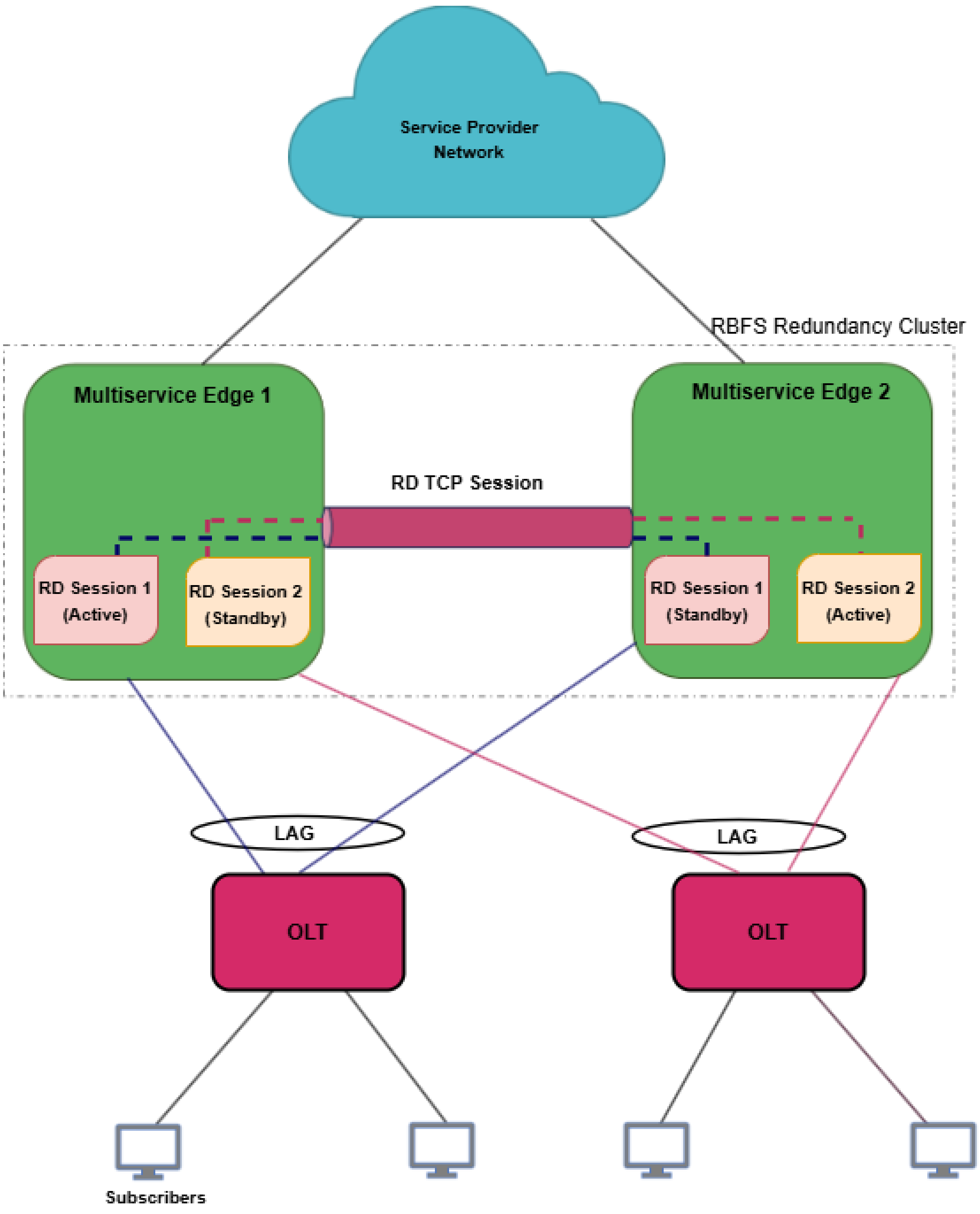
If the session goes down, the Keepalive messages cannot be exchanged between the Multiservice Edge nodes and the Multiservice Edge nodes move to the standalone state only after the hold-timer expires and the previously synchronized subscriber session data becomes invalid.

If the Multiservice Edge pairs are connected through the core network, then a core network failure may impact redundancy and trigger a false down event. However, if the inter-BNG connection is direct or through a different path that does not rely on the core, redundancy is not impacted by core network failure.

In a scenario, when a new lag is configured for redundancy with the same redundancy session ID, it works for the same session. If any of the lag goes down, a switchover happens.

IPoE Stateful Redundancy Architecture

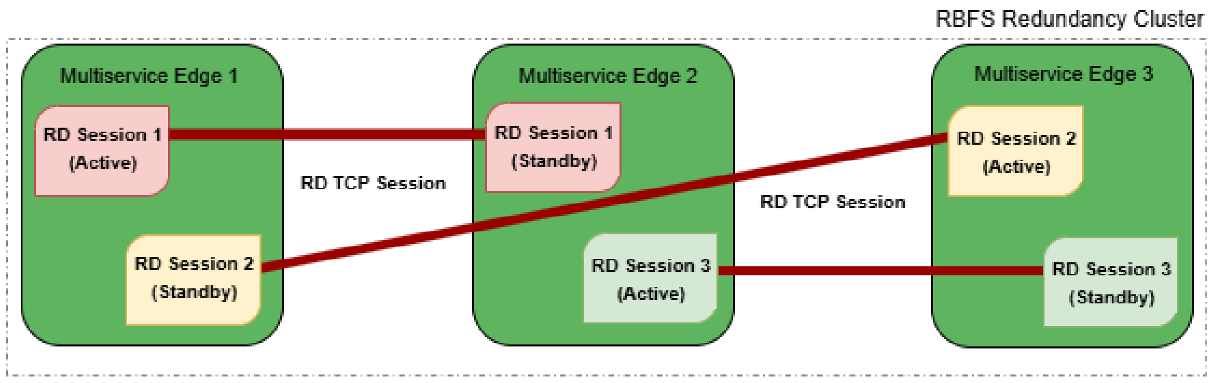
The following architectural diagram provides a high-level view of IPoE Stateful redundancy mode. It shows two RBFS nodes, paired for redundancy, deployed in an active-standby node cluster, with their interfaces are connected with an RD TCP connection. These peer nodes use the RD TCP connection for sending 'keepalive' messages and data mirroring for subscriber state synchronization.



Both of the nodes are connected to an access device (OLT device in this scenario) on one end from where it receives subscriber traffic and sends traffic. The nodes are also connected to the core network on the other end.

The node 'Multiservice Edge 1' is in active state and performs subscriber services for the 'Session 1' (redundancy subscriber group). 'Session 1' is also mirrored in the Multiservice Edge 2 in standby mode. The standby device mirrors concurrent subscriber state data for 'Session 1' from the active peer. If an active node goes down due to any reason, the peer node detects the outage and uses mirrored 'Session 1' to perform subscriber services.

One Multiservice Edge node acts as an active node for one or more sessions (subscriber redundancy groups) and as a standby Multiservice Edge for other subscriber redundancy groups at the same time. The following diagram illustrates the scenario.



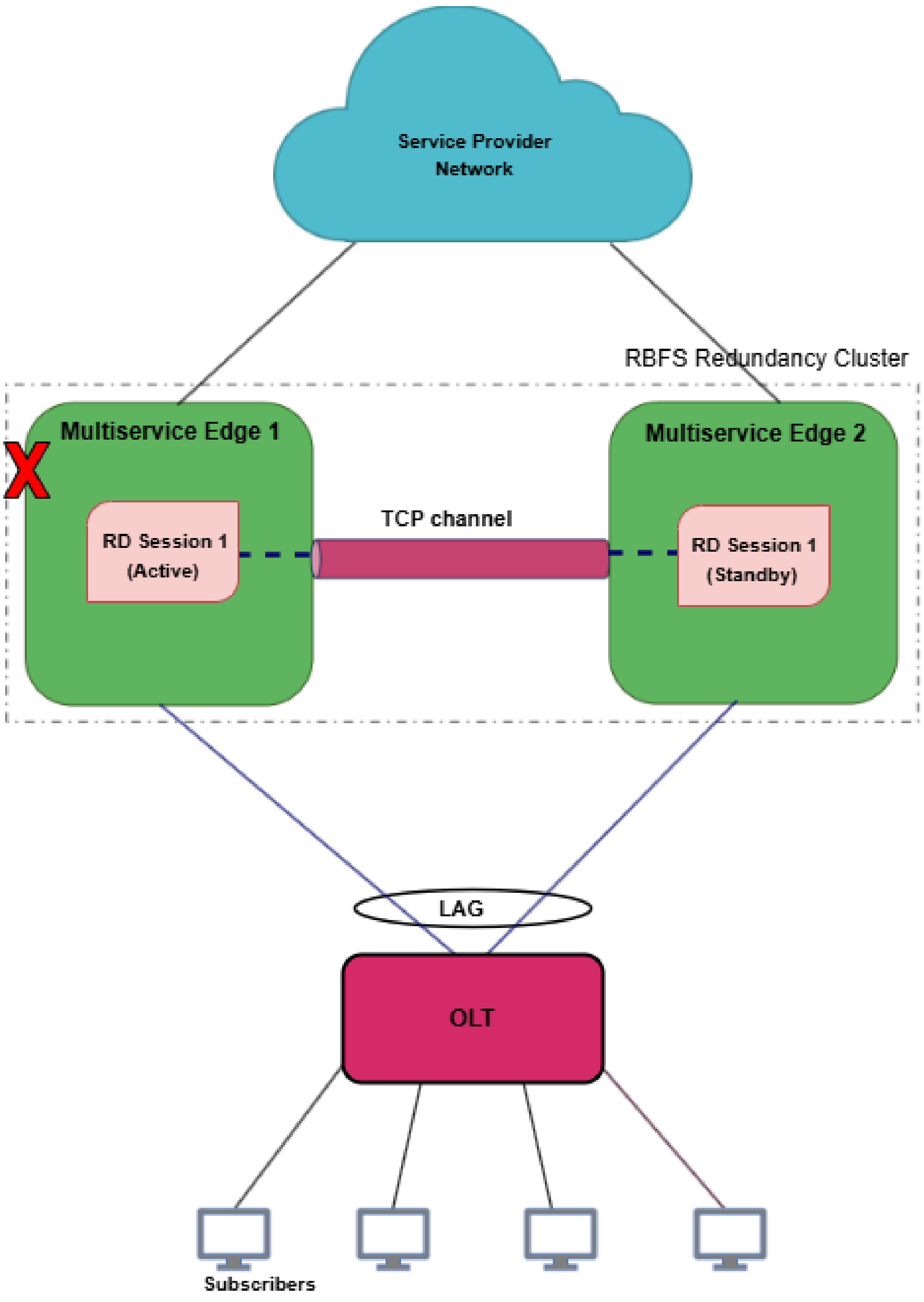
'Session 1' is active in Multiservice Edge 1 and is in standby mode in Multiservice Edge 2.
'Session 2' is active in Multiservice Edge 3 and is in standby mode in Multiservice Edge 1.
'Session 3' is active in Multiservice Edge 2 and is standby mode in Multiservice Edge 3.

IPoE Stateful Redundancy can mitigate the following types of failures:

- Link failure Between Active RBFS Node and Access Node (OLT, DSLAM or MSAN)
- Node Outage

Redundancy for Node Outage

A node outage, which can bring down the subscriber services, can occur due to many reasons on a network. IPoE Stateful Redundancy helps to minimize the impact and reduce interruptions and downtime by providing a resilient system. In the event of a node outage, IPoE Stateful Redundancy triggers switchover in which the standby node takes over from the active node with very minimal impact on the subscriber services.

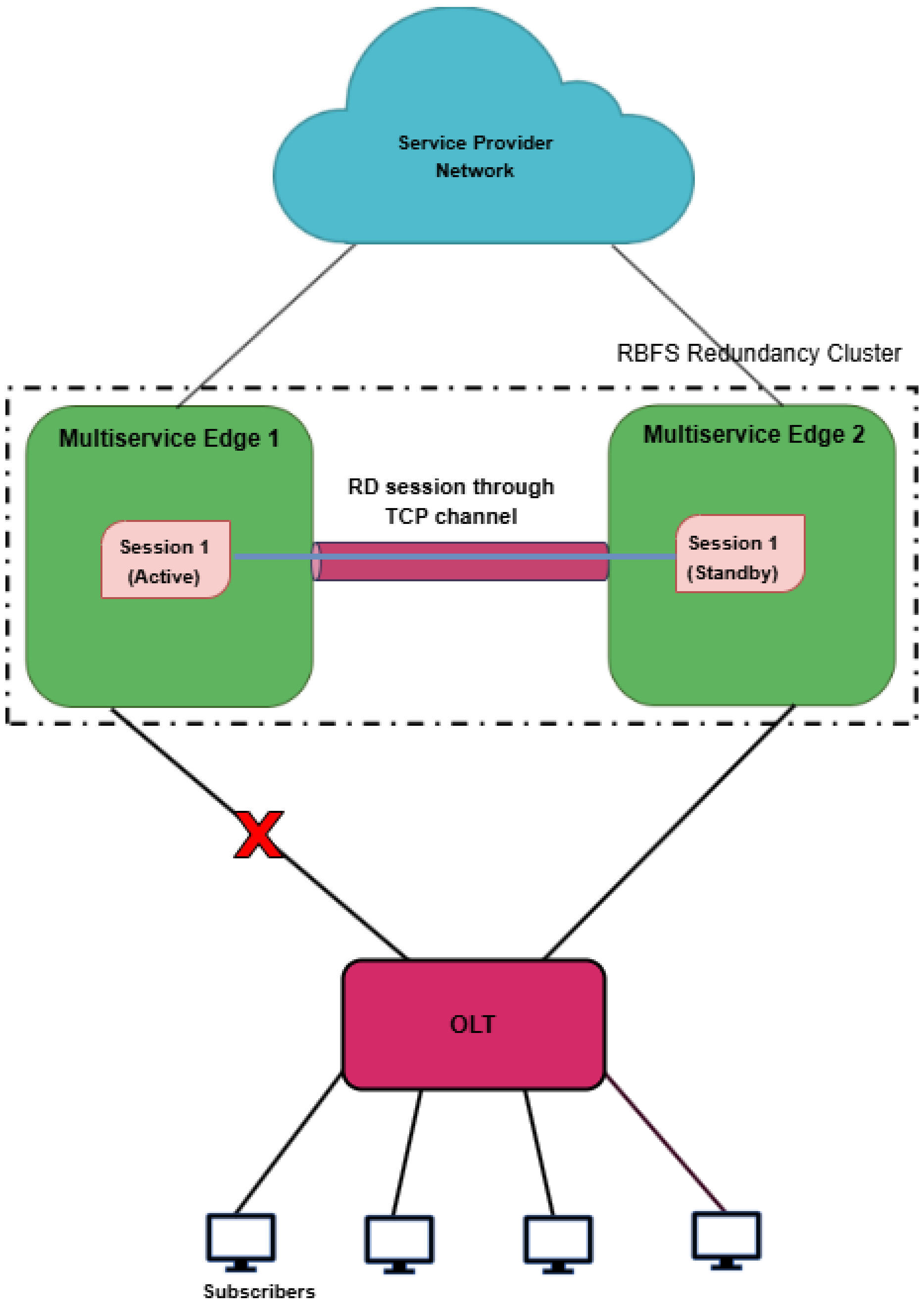


The diagram shows multiservice-edge1 as an active node serving subscribers and multiservice-edge2 stays as a standby. When an active node goes down, the standby node detects the same and takes over from the active RBFS node. In a node outage scenario, the node becomes unresponsive and cannot maintain communication with its peer node the RD TCP link.

Redundancy for Link Failure

In IPoE Stateful Redundancy, Link Aggregation combines multiple physical links into a single logical link. If a member link, which is part of a LAG, goes down, the

LAG fails. The diagram shows the 'multiservice-edge1' and 'multiservice-edge2' deployed in redundancy mode and is connected to the access device with LAGs. When the LAG between 'multiservice-edge1' and the access node goes down, the 'multiservice-edge1', which is running 'Session 1', becomes inactive for the subscriber group. So the 'multiservice-edge2', which is the standby for 'Session 1', detects the failure of 'multiservice-edge1' and starts performing subscriber services for 'Session 1' by providing a quick recovery from the disruption.



In this link or LAG failure scenario, multiservice-edge1 is in a healthy state, only the LAG interface went down. It cannot perform subscriber services for the particular redundancy session. However, it can keep communication with its peer node as the RD TCP channel.

Node States in Redundancy

In IPoE Stateful redundancy, Multiservice Edge nodes have different states for various redundancy sessions. Typically, IPoE Stateful redundancy nodes encounter the following states for redundancy sessions:

Active: All subscribers are served by active node in the RBFS active-standby node cluster. One node which is active for a redundancy session can be a standby node for a different session. Nodes, that are paired for redundancy, send 'keepalive' messages to each other and also synchronize all subscriber state data with the peer node. The priority values that you specify for the redundancy nodes determine the roles of active and standby. The node that receives the higher priority value for the session ID assumes the role of active for that subscriber group. To set one device as 'active', you must specify higher priority value for the redundancy session for that node.

Standby: Standby node is identical with the active node and synchronizes subscriber data concurrently from peer node. It keeps communication with the peer node to monitor node health status using 'keepalive' messages. The node that gets the lower priority value for the redundancy session ID assumes the role of standby. Standby node for a subscriber group does not perform any subscriber services for that group unless or until the active node encounters an outage.

Down: When a node becomes inactive due to an outage, it is considered as 'down'. In the event of a node outage, it is completely down and cannot perform subscriber services and any communication with its peer node. But in the case of a LAG (between the node and access node) failure, the node cannot perform subscriber services, but it can communicate with the peer node through the RD TCP connection. So that the subscriber state synchronization occurs without any interruption.

Stand Alone: When the active node goes down, the switchover occurs and standby takes over the subscriber service. In this scenario, the serving node is in 'stand alone' state (for that redundancy session) as it has no peer node for redundancy.

Revert or Rollback Active-Standby Redundancy Mode

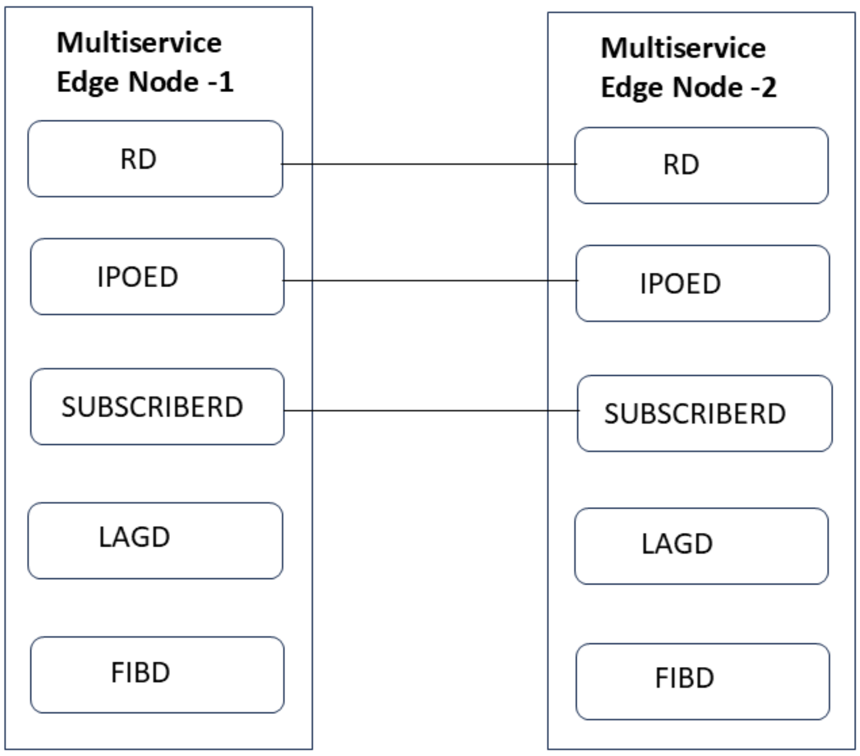
In IPoE Stateful Redundancy, after a node or link failure and the subsequent switchover, the standby takes over and continues the subscriber service for that subscriber group even after the other node (previously active) recovers from the failure. There is no automated rollback or revert to the previously active router. However, administrators can perform a manual switchover for the Active-Standby Redundancy mode.

Monitoring Node Health Status

The RBFS nodes, which have switchover capacities, monitor each other for the health status. IPoE Stateful Redundancy uses 'keepalive' messages that check on the health of the RBFS nodes. Both of the devices send 'keepalive' messages to each other in every five seconds. One Node can detect a failure if it does not receive 'keepalive' messages for a period of 20 seconds from the other node.

Redundancy Clients

There are multiple RBFS daemons that participate in providing redundancy. They include redundancy daemon, (rd), LAG daemon (lagd), interface daemon (ifmd), subscriber daemon (subscriberd), IPoE daemon (ipoed) and forwarding daemon (fibd). These daemons, which perform various roles, are known as redundancy clients.



Redundancy Daemon

Redundancy Daemon is responsible for establishing high-availability connections. It monitors the ecosystem and detects any outages that may happen on the network. It assigns the roles of active or standby to the nodes depending on the priority configured on the node. The daemon triggers a switchover to the standby node if a failure occurs. It responds to the failure events which are reported locally by daemons who are the redundancy clients. It also reconciles the data after switchover from the node that went down.

Supported Hardware Platforms

Currently, RBFS can be deployed in redundancy mode using the RBFS Multiservice Edge switches. Multiservice Edge software provides complete BNG functionalities on a single compact hardware switch. Both the Active-Active Redundancy and Active-Standby Redundancy modes are supported in the following hardware platforms:

- UfiSpace S9600-72XC: The UfiSpace S9600-72XC is a multi-function, disaggregated white box aggregation routing platform that is equipped with Broadcom’s Qumran2c chipset. It features 64x25GE and 8x100GE high-speed ports with a switching capacity of up to 2.4Tbs.

- Edgecore AGR420: AGR420 is a high performance 25GbE aggregation router that consists of fixed 64 x 10G/25G SFP28, 8 x 100GE QSFP28 and 2 x 100G QSFP-DD network interface configurations.

Only Active-Standby Mode is supported on the following platforms:

- Edgecore CSR440
- UfiSpace S9510-28DC

IPoE Stateful Redundancy Requirements

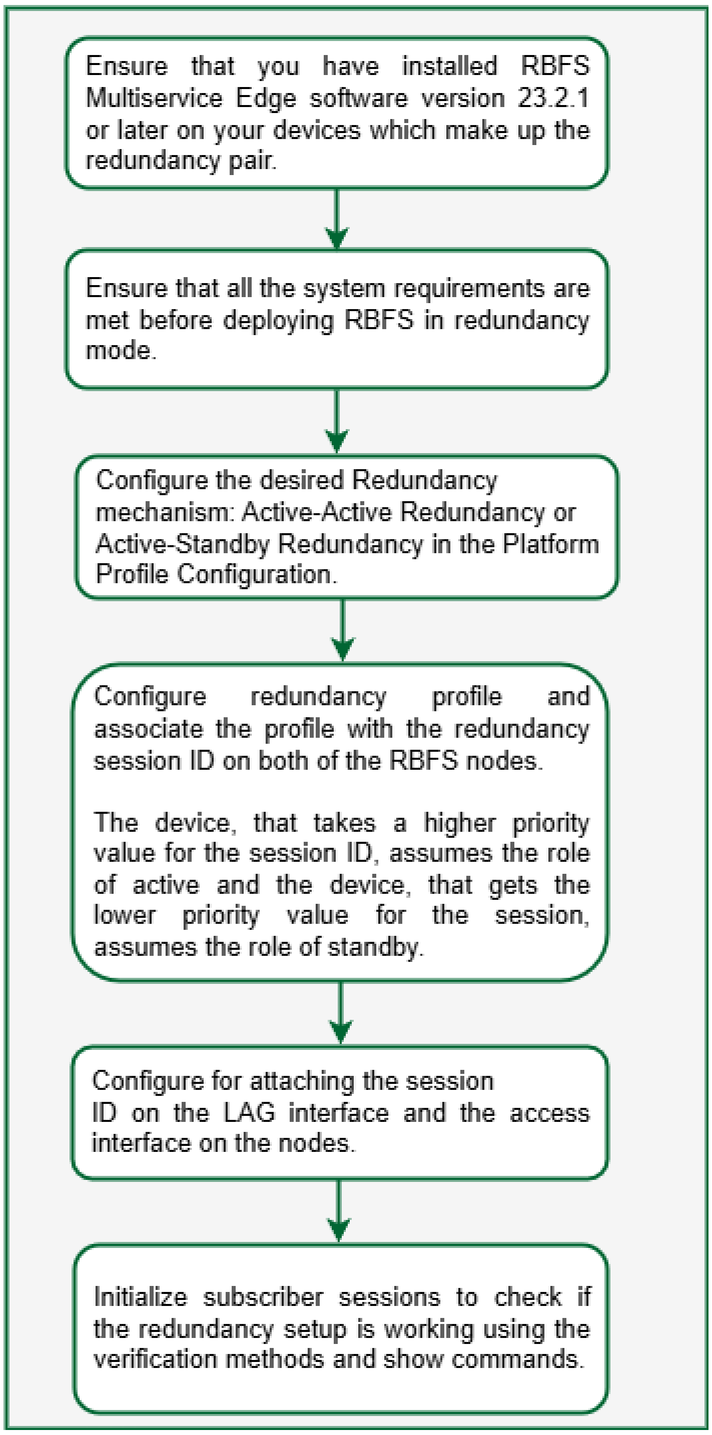
The following are the requirements for setting up Redundancy.

- Ensure that both of the platform devices, on which RBFS software runs, must be the same model.
- Ensure that the devices should run the same version of RBFS software. RBFS software 23.2.1 and later versions support Active-Standby redundancy mode and 25.1.1 and later versions support Active-Active Redundancy mode.
- NTP must be configured on both devices to match the timestamps.

5.9.2. Deploying Multiservice Edge in Redundancy Mode

Redundancy

This section provides information on how to deploy Multiservice Edge device pair to achieve redundancy. The following workflow diagram depicts the end-to-end deployment of Multiservice Edge in Redundancy mode:


Ensure that you have installed RBFS Multiservice Edge software version 23.2.1 or later on your devices which make up the redundancy pair.
Ensure that all the system requirements are met before deploying RBFS in redundancy mode.
Configure the desired Redundancy mechanism: Active-Active Redundancy or Active-Standby Redundancy in the Platform Profile Configuration.
Configure redundancy profile and associate the profile with the redundancy session ID on both of the RBFS nodes.
The device, that takes a higher priority value for the session ID, assumes the role of active and the device, that gets the lower priority value for the session, assumes the role of standby.
Configure for attaching the session ID on the LAG interface and the access interface on the nodes.
Initialize subscriber sessions to check if the redundancy setup is working using the verification methods and show commands.

You must perform the following tasks to deploy the RBFS devices in redundancy mode. These configurations must be performed on both of the devices.

1. Set up the Redundancy Mode: Standby Forwarding or Standby Non-Forwarding.
2. Configure Redundancy Profile
3. Configure Session for Redundancy
4. Configure Link Aggregation Group for Redundancy

5. Configure Access for Redundancy

Configuration Syntax and Commands

The following sections describe syntax and commands for various configurations.

Setting up the Redundancy Mode

You can choose your preferred redundancy mechanism and set up the system to enable it. Redundancy is available in two modes: Standby Forwarding and Standby Non-Forwarding. By default, the Standby Forwarding method is enabled. If you would like to use the Standby Non-Forwarding mode instead, you need to select the Standby Non-Forwarding option in the Platform Profile configuration.

This setup is a one-time activity and system reboot is required the changes to take effect.

To set the desired redundancy mode, use the command.

Syntax:

set system platform profile <profile-name> **ipoe-mode** [**Standby-Forwarding** | **Standby-Non-Forwarding**]

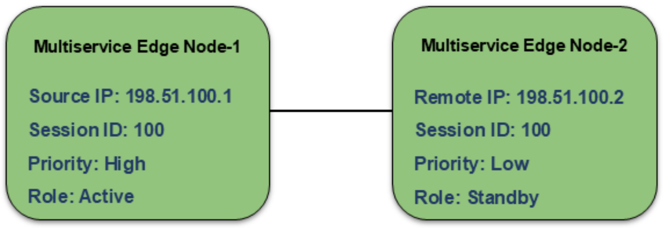
Attribute	Description
standby-forwarding	Active-Active redundancy mode. Redundancy mode in which the standby LAG is utilized for traffic forwarding.
standby-non-forwarding	Active-Standby Redundancy mode. Redundancy mode in which Standby node LAG does not forward traffic unless a failover occurs.

Example:

```
supervisor@rtbrick.net: cfg> show config system platform profile
{
  "rtbrick-config:profile": [
    {
      "profile_name": "nat_4q",
      "ipoe-mode": "standby-non-forwarding"
    }
  ]
}
```

Configuring Redundancy Profile

Redundancy profile configuration is used to provide peer identity in redundancy. While configuring the redundancy profile on the nodes, you need to specify IP addresses of both the peer nodes. Based on the priority value that you specify for the session ID, the peers take the roles of active and standby for the Session.



Syntax:

set redundancy profile <name>

Attribute	**Description**
peer	Redundancy configuration
switchover-hold-timer	Minimum time interval between consecutive switch-overs in seconds.

Run the following commands to configure redundancy profile.

```
set redundancy profile rd_ipoe
set redundancy profile rd_ipoe peer ipv4 remote-address 198.51.100.2
set redundancy profile rd_ipoe peer ipv4 update-source 198.51.100.1
set redundancy profile rd_ipoe peer ipv4 instance default
```

Example Configuration:

```
supervisor@rtbrick>cbng1.rtbrick.net: cfg> show config redundancy profile
{
    "rtbrick-config:profile": [
      {
        "name": "rd_ipoe",
        "peer": {
          "ipv4": {
            "remote-address": "198.51.100.2",
            "update-source": "198.51.100.1",
            "instance": "default"
          }
        }
      }
    ]
```

```
  }
```

Configuring Session for Redundancy

You can configure Redundancy Session with a unique session ID. You can define the system priority value (which determines active and standby roles) and the associate Redundancy profile configuration in the Redundancy Session configuration.

Syntax:

set redundancy session <session-id>

Attribute	Description
keepalive-interval	Keepalive message transmission interval in seconds. Default is 5 seconds.
priority	Session priority
profile	Profile name

Run the following commands to configure session for redundancy.

```
set redundancy session 100
set redundancy session 100 priority 10
set redundancy session 100 profile rd_ipoe
```

Configuration Example:

```
supervisor@rtbrick>cbng2.rtbrick.net: cfg> show config redundancy session
{
    "rtbrick-config:session": [
      {
        "session-id": 100,
        "priority": 10,
        "profile": "rd_ipoe"
      }
    ]
  }
```

Configuring LAG for Redundancy

You must associate the Redundancy Session with the LAG. While configuring LAG for redundancy, you must specify the session ID to associate the LAG with the Redundancy Session. LAG can identify its Redundancy Session with this mapping.

Syntax:

set link-aggregation interface <interface-name> **options**

Attribute	Description
description	Link aggregation interface description.
member-interface	Link aggregation member interface configuration
minimum-link-count	Minimum number of active member links required for the link aggregation interface. default values is 1.
mode	Mode of the link aggregation interface, static or lacp. default mode is lacp.
rd-role	Role of the link aggregation interface, active or standby.
rd-system-priority	The value for the system priority range from 1 to 65535. The lower the value, the higher the system priority. default value is 65535.
redundancy-session-id	The value for the redundancy group session id range from 1 to 65535.
system-id	Redundancy System ID of link-aggregation interface.

Run the following commands to configure LAG for redundancy.

```
set link-aggregation interface lag-1
set link-aggregation interface lag-1 mode lacp
set link-aggregation interface lag-1 minimum-link-count 1
set link-aggregation interface lag-1 redundancy-session-id 100
set link-aggregation interface lag-1 system-id a8:b5:7e:8f:66:43
set link-aggregation interface lag-1 member-interface ifp-0/1/260
```

Example Configuration:

```
supervisor@rtbrick>cbng2.rtbrick.net:: cfg> show config link-aggregation interface
lag-1
{
    "rtbrick-config:interface": [
      {
        "interface-name": "lag-1",
        "mode": "lacp",
        "minimum-link-count": 1,
        "redundancy-session-id": 100,
        "system-id": "11:22:33:44:55:66",
        "member-interface": [
          {
            "member-interface-name": "ifp-0/0/4"
          }
        ]
      }
    ]
  }
```

Configuring Access for Redundancy

IPoE and Redundancy Session mapping is essential to associate the Redundancy Session with IPoE. While configuring access for redundancy, you must specify access type as IPoE.

Syntax:

set access interface double-tagged <name> <options>

Attribute	Description
aaa-profile-name	AAA profile name
access-profile-name	Access profile name
access-type	Access service type
gateway-ifl	IPoE gateway IFL (unnumbered source IFL)
max-subscribers-per-mac	Restrict maximum subscribers per MAC address
max-subscribers-per-vlan	Restrict maximum subscribers per VLAN
redundancy-session-id	Redundancy session id for this interface
service-profile-name	Service profile name
vlan-profile-enable	Enable VLAN profiles

Run the following commands access for redundancy.

```
set access interface double-tagged lag-1 1001 1100 1001 1100
set access interface double-tagged lag-1 1001 1100 1001 1100 access-type IPoE
set access interface double-tagged lag-1 1001 1100 1001 1100 access-profile-name
ipoe
set access interface double-tagged lag-1 1001 1100 1001 1100 aaa-profile-name
ipoe-aaa
set access interface double-tagged lag-1 1001 1100 1001 1100 gateway-ifl lo-
0/0/0/10
set access interface double-tagged lag-1 1001 1100 1001 1100 redundancy-session-id
100
```

Example Configuration:

```
supervisor@rtbrick>cbng1.rtbrick.net: cfg> show config access interface double-
tagged lag-1
{
    "rtbrick-config:double-tagged": [
      {
        "interface-name": "lag-1",
        "outer-vlan-min": 1001,
        "outer-vlan-max": 1100,
        "inner-vlan-min": 1001,
        "inner-vlan-max": 1100,
        "access-type": "IPoE",
        "access-profile-name": "ipoe",
        "aaa-profile-name": "ipoe-aaa",
        "gateway-ifl": "lo-0/0/0/10",
        "redundancy-session-id": 100
      }
    ]
  }
```

Switchover Manually

In the Active-Standby redundancy mode, administrators can perform a manual switchover from an active node to a standby node. Use the following command to perform switchover:

```
switch-over session <session-id> confirm
```

5.9.3. RBFS Redundancy Operational Commands

Redundancy Show Commands

With the RBFS Command Line Interface, you can view output of operational commands. The redundancy operational commands provide detailed information about the RBFS redundancy operations.

Redundancy Details

Syntax:

show redundancy <options>

Option	Description
client	Displays information about redundancy client daemons.
session	Displays session details.

Example: Details for a specified client daemon.

```
supervisor@rtbrick.net: op> show redundancy client lagd statistics
Session id: 100, Profile: rd_ipoe_olt1
  Agent State: active
  TCP operational state: up
  Message statistics:
    Keepalive sent    : 0
    Keepalive received: 0
  Last 5 state changes: [Latest first]
    active              : 2025-03-14T09:40:00.042489+0000
    promote-ready       : 2025-03-14T09:40:00.042460+0000
    promote-infra-wait  : 2025-03-14T09:40:00.042435+0000
    promote-app-wait    : 2025-03-14T09:40:00.041832+0000
    down                : 2025-03-14T09:39:45.931660+0000
  Connection statistics:
    Peer Address: 192.1.0.200
    Application down notifications: 0
    Connection down notifications: 0
    Retry count: 0
    Session down received: 0
Session id: 200, Profile: rd_ipoe_olt2
  Agent State: standby
  TCP operational state: up
  Message statistics:
    Keepalive sent    : 0
    Keepalive received: 0
  Last 5 state changes: [Latest first]
    standby             : 2025-03-14T09:40:01.025264+0000
    demote-ready        : 2025-03-14T09:40:01.025249+0000
    demote-infra-wait   : 2025-03-14T09:40:01.025229+0000
    demote-app-wait     : 2025-03-14T09:40:01.025197+0000
    down                : 2025-03-14T09:39:45.932673+0000
  Connection statistics:
    Peer Address: 192.1.0.20
```

Example: Details for a specified session ID

```
supervisor@rtbrick.net: op> show redundancy session 100
Instance: default
  Session ID     Interface   Peer        Source     State     Peer state  Keepalive
rcvd Keepalive sent  Mode
  100         lag-1       192.1.0.200 192.1.0.100 active standby      563
564      standby-forwarding
```

Example: Details for redundancy session details

```
supervisor@M1-STD-44-4401>bm13-tst.fsn.rtbrick.net: op> show redundancy session
detail
Redundancy session ID: 100, Source: 192.1.0.100, Peer: 192.1.0.200
  Instance: default, Profile name: rd_ipoe_olt1, Local priority: 20
  State: active, Previous state: connect, Last state transition time: 2025-03-
14T09:40:00.034321+0000
  Peer state: standby
  Mode: standby-forwarding
  Interface: lag-1
  TCP operational state: up
  Message statistics:
    Keepalive sent       : 569
    Keepalive received   : 568
    Keepalive ignored    : 0
  Switchovers via cli    : 0
  State transition statistics:
    Down               : 0
    Connect            : 1
    Standby            : 0
    Active             : 1
    Stand-alone        : 0
    Demote             : 0
  Timers:
    Connect retry interval               : 4294967295
    Keepalive timer interval             : 5000
    Holddown timer interval              : 20000
    Switchover timer pending interval    : 0 sec
  Timer Statistics:
    Holddown timer expiry count          : 0
Redundancy session ID: 200, Source: 192.1.0.10, Peer: 192.1.0.20
  Instance: default, Profile name: rd_ipoe_olt2, Local priority: 210
  State: standby, Previous state: connect, Last state transition time: 2025-03-
14T09:40:01.019764+0000
  Peer state: active
  Mode: standby-forwarding
  Interface: lag-11
:
```

Client Statistics

Syntax:

show redundancy client <name> **statistics**

This command displays information from redundancy client (daemon) which is a participant in the redundancy sessions. The daemons ifmd, ipoed.1, lagd, and subscriberd.1 are the client daemons in redundancy sessions.

Command:

```
show redundancy client lagd statistics
```

Example:

```
supervisor@rtbrick>ufi08.q2c.u23.r4.nbg.rtbrick.net: op> show redundancy client
lagd statistics
Session id: 100, Profile: rd_ipoe
  Agent State: down
  TCP operational state: down
  Message statistics:
    Keep alive sent: 0
    Keep alive received: 0
  Last 5 state changes: [Latest first]
    down                : 2022-12-08T08:47:06.077823+0000
    demote-ready        : 2022-12-08T08:47:06.077798+0000
    demote-infra-wait   : 2022-12-08T08:47:06.077769+0000
    demote-app-wait     : 2022-12-08T08:47:06.077713+0000
    active              : 2022-12-08T06:57:53.058172+0000
  Connection statistics:
    Peer Address: 198.51.100.2
    Application down notifications: 0
    Connection down notifications: 0
    Retry count: 0
    Session down received: 8
```

Redundancy Session Details

Syntax:

show redundancy session <ID> <options>

Option	Description
-	Without any option, it displays all session details.
detail	Displays details session information.
history	Displays session history.
status	Displays session status.

Example:

```
supervisor@rtbrick.net: op> show redundancy session detail
Redundancy session ID: 100, Update source: 198.51.100.1, Peer: 198.51.100.2.
  Instance: default, Profile name: rd_ipoe, Local priority: 20
  State: down, Previous state: active, Last state transition time: 2022-12-
08T08:47:06.071930+0000
  TCP operational state: down
  Message statistics:
    Keep alive sent: 21516
    Keep alive received: 21515
    Switch overs detected: 0
  Timers:
    Connect retry interval: 2000
    keep alive timer interval: 3000
    Holddown timer interval: 9000
```

Example: RD session detail for a session ID.

```
supervisor@rtbrick.net: op> show redundancy session 100 detail
Redundancy session ID: 100, Update source: 198.51.100.1, Peer: 198.51.100.2
  Instance: default, Profile name: rd_ipoe, Local priority: 20
  State: down, Previous state: active, Last state transition time: 2022-12-
08T08:47:06.071930+0000
  TCP operational state: down
  Message statistics:
    Keep alive sent: 21516
    Keep alive received: 21515
    Switch overs detected: 0
  Timers:
    Connect retry interval: 2000
    keep alive timer interval: 3000
    Holddown timer interval: 9000
```

Example: Status information of a session specified

```
supervisor@rtbrick.net: op> show redundancy session 100 status
State: down,  Remote State: invalid
Redundancy client replication information:
  Total redundancy clients : 5
  ifmd:
  ipoed.1:
  lagd:
  poold:
  subscriberd.1:
    Number of subscribed table: 1
```

Example: History of a session ID for a specified count.

```
supervisor@rtbrick.net: cfg> show redundancy session 100 history count 3
Previous state       Current state        State change reason  Timestamp
connect              standby              standby              2022-12-
21T07:05:36.010847+0000
down                 connect              open                 2022-12-
21T07:05:29.738121+0000
invalid              down                 init                 2022-12-
21T07:00:44.282300+0000
```

6. Forwarding

6.1. Interfaces

6.1.1. Interfaces Overview

RtBrick Full Stack (RBFS) supports various types of interfaces, including physical and logical interfaces. On hardware platforms, RBFS physical interfaces represent the ports of a switch. This guide describes how to configure and verify RBFS interfaces. Features like routing protocols or access services will typically run on top of the interfaces.

Interface Types

Physical Interfaces

In RBFS, physical interfaces (IFP) typically represent the physical ports of a hardware switch. For example, ifp-0/0/1 represents switch port 1. On the physical interface level, you can configure various parameters associated with Layer 1 of the ISO/OSI reference model.

Logical Interfaces

For each physical interface, you can create one or multiple interface units also referred to as logical interfaces (IFL) in RBFS. A logical interface is associated with the Layer 2 operation. In addition, you can configure Layer 3 parameters like IP addresses on interface units, and assign interface units to routing instances.

Loopback Interfaces

A loopback interface is typically used to represent and identify a device itself. Loopback interfaces are preferred because they do not depend on the status of a physical port, and will always be up. Please note, although loopback interfaces are virtual interfaces, there are also represented as physical interfaces and interface units in RBFS, reflecting Layer 1 and Layer 2/3 operation.

Host Interfaces

Linux virtual ethernet (veth) interfaces connect an LXC container with the Linux host OS. In RBFS, a veth interface to the Linux bridge lxcbr0 is created by default. In virtual topologies, you can create additional veth interfaces and

Linux bridges. RBFS host interfaces represent veth interfaces in RBFS.
For example, if the container interface eth1 connects to the host interface vethXYZ123, ifp-0/0/1 can be bound to eth1 to represent it in RBFS. Host interfaces can be used like any other physical interface.

Memory Interfaces

Memory interfaces (memif) are virtual interfaces used for creating virtual topologies. They connect multiple containers running RBFS to each other. When configuring memif interfaces:

- Endpoints match on the memif ID, i.e. the memif ID needs to be the same on both ends.
- memif IDs need to be unique on the host.
- The memif interface name is locally significant only.
- One endpoint needs to be configured as a master, while the other one is configured as a slave.

Interface Numbering

RBFS interface numbers match the port numbers on the switch faceplate. An interface is named in the ifp-<chassis-ID>/<front-panel-block-number>/<port> format. For example, ifp-0/0/1.

- Chassis ID—always 0 for the currently supported platforms
- Front Panel Block—represents group of ports on the faceplate
- Port—matches the port number on switch faceplate

Virtual interfaces follow the same structure, for example, lo-0/0/1 or memif-0/0/1.

Logical interfaces are numbered: ifl-<Node ID>/<Chip ID>/<Port ID>/<Unit ID>, for example ifl-0/0/1/1.

Community Support for Interfaces

You can tag an interface address with a community or extended community. RBFS will create a direct route for each interface address. If a community or extended community is configured for an interface address, RBFS will add it to the direct route. Communities can be used in policies. For example, when redistributing direct routes, you can match these communities and define desired policy rules.

Unnumbered Interfaces

An unnumbered interface is a point-to-point interface that is not explicitly configured with a dedicated IP address and subnet. Instead, it borrows (or links to) from a loopback interface, and uses it as the source IP address for packets originating from the interface. The IP unnumbered interface can "borrow" the IP address from another interface that is already configured on the switch, thereby conserving network and address space.

The IP address of the unnumbered interface cannot be borrowed as it has no dedicated IP address. A logical interface can borrow IP address from a loopback interface, not vice versa.

RBFS supports using /32 subnet masks for unnumbered interfaces and loopback interfaces.

Unnumbered IFL to Numbered IFL (and Vice versa) Conversion

RBFS supports the conversion of unnumbered to numbered IFL and vice versa in a single commit. The existing address should be deleted before converting the IFL to an unnumbered IFL. These two operations are supported in one commit.

The examples below show the behavior of converting unnumbered to numbered IFL and vice versa within a single commit in both the previous and current releases of RBFS.

Updated CLI Behavior for Numbered to Unnumbered Interface

```
supervisor@rtbrick>: cfg> delete interface ifl-0/0/0 unit 10 address ipv4 10.1.1.1/24
supervisor@rtbrick>: cfg> set interface ifl-0/0/0 unit 10 unnumbered interface lo-0/0/1/0
supervisor@rtbrick>: cfg> commit
```

Updated CLI Behavior for Unnumbered to Numbered Interface

```
supervisor@rtbrick>: cfg> del interface ifl-0/0/3 unit 10 unnumbered interface lo-0/0/1/0
supervisor@rtbrick>: cfg> set interface ifl-0/0/3 unit 10 address ipv4 10.1.1.1/24
supervisor@rtbrick>: cfg> commit
```

Interface States

RBFS supports various interfaces such as physical interface, logical interface, and LAG interface and it uses various indicators to show the various states of interfaces. All interfaces have different states such as the following:

- Admin state: Indicates whether the interface is enabled (Up) or disabled (Down).
- Link state: Indicates whether the interface is linked (Up) or not linked (Down).
- Oper state: Indicates whether the interface is functional (operational) or not functional.
- IPv4 state: Indicates that the interface is configured with the IPv4 address.
- IPv6 state: Indicates that the interface is configured with the IPv6 address.
- MPLS state: Indicates that the interface is an MPLS-enabled interface.

Physical Interface States:

The physical interface states include:

- admin
- oper (operational)
- link

Logical Interface States:

The logical interface states include:

- admin
- oper (operational)
- link
- IPv4
- IPv6
- MPLS

LAG Interface States

The LAG interface states include:

- admin
- oper(operational)

The interface state can either be Up or Down. Up shows that it is ready to pass packets, and Down shows that it is not ready to transmit packets. The operational state shows that the interface is operational and is ready to transmit packets. The Oper state is Up only if both the Admin state and Link state are Up.

Auto-negotiation

Auto-negotiation allows directly connected devices to automatically exchange speed and duplex mode information for the links. If auto-negotiation is enabled, ports can auto-negotiate the speed and duplex capabilities with other ports. The auto-negotiation can determine the best speed and duplex at which the ports can operate optimally.

Port speed configuration and auto-negotiation are mutually exclusive.

RBFS supports auto-negotiation between ports in the following ways:

- 1G ports can negotiate with 10G ports.
- 40G ports can negotiate with 100G ports.

Auto-negotiation is not supported for the following combinations:

- 40G ports cannot negotiate with 1G ports, 10G ports, and 25G ports.
- 100G ports cannot negotiate with 1G ports, 10G ports and 25G ports.
- 1G port cannot negotiate with 25G port.

Interface Counters

Interface counters are statistics that network devices maintain for the traffic passing through the interfaces for all physical, logical, and LAG interfaces on a device. These counters provide information about the utilization and performance of the interface. Users can identify and troubleshoot issues such as congestion, errors, and performance bottlenecks by monitoring these counters.

RBFS Logical Interface counters and descriptions

Counters	**Descriptions**
Rx packets	The number of IP packets received by the logical interface.
Rx bytes	The number of bytes received by the logical interface.
Tx packets	The number of packets transmitted by the logical interface.
Tx bytes	The number of bytes transmitted by a logical interface.
IPv4 packets	The number of IPv4 packets processed by a logical interface.
IPv6 packets	The number of IPv6 packets processed by a logical interface.
MPLS packets	The number of MPLS packets processed by a logical interface.
Punt packets	The number of packets that are punted or forwarded to the CPU for further processing by a logical interface.
Drops packets	The number of packets that were dropped by a logical interface.
Rx Miss packets	The number of packets that were dropped by a logical interface.
Rx Error packets	The number of packets that were received with errors by the logical interface.
Rx No Buff packets	The number of packets that could not be received due to a lack of buffer space. When a logical interface receives a packet but does not have enough buffer space to store it, the packet is dropped.
Tx Error packets	The number of packets that could not be transmitted due to errors encountered during the transmission process.
Packet Statistics	
Ingress forwarded packets	The number of packets that are received by the device on one interface and then forwarded out on another interface.

Counters	Descriptions
Ingress forwarded bytes	The number of bytes that are received by a device on one interface and then forwarded out on another interface.
Ingress drop Packets	The number of packets that are dropped (discarded) by a network device upon arrival on one of its interfaces by a device.
Ingress drop bytes	Total number of bytes that have been dropped by an interface on incoming traffic.
Egress forwarded packets	Total number of packets that have been successfully forwarded by an interface on outgoing traffic.
Egress forwarded bytes	The number of bytes that have been forwarded by an interface in the egress direction.
Egress drop packets	The number of packets that have been dropped by an interface in the egress (outgoing) direction.
Egress drop bytes	The number of bytes that have been dropped by an interface in the egress direction.

RBFS physical interface counters and descriptions

Physical Interface Counters	Descriptions
VPP Statistics	
Rx packets	The number of IP packets received by the physical interface.
Rx bytes	The number of bytes received by the physical interface.
Tx packets	The number of packets transmitted by the physical interface.
Tx bytes	The number of bytes transmitted by the physical interface.
IPv4 packets	The number of IPv4 packets processed by the physical interface.

Physical Interface Counters	Descriptions
IPv6 packets	The number of IPv6 packets processed by the physical interface.
MPLS packets	The number of MPLS packets processed by a physical interface.
Punt packets	The number of packets that are punted or forwarded to the CPU for further processing by a physical interface.
Drops packets	The number of packets that were dropped by a physical interface.
Rx Miss packets	The number of packets that were dropped by a physical interface.
Rx Error packets	The number of packets that were received with errors by the physical interface.
Rx No Buff packets	The number of packets that could not be received due to a lack of buffer space. When a physical interface receives a packet but does not have enough buffer space to store it, the packet is dropped.
Tx Error packets	The number of packets that were dropped by a physical interface.
BCM Statistics	
inOctets	Number of octets received through the interface.
inUcastPkts	Number of unicast packets received through the interface.
inNonUcastPkts	Number of non-unicast packets received through the interface.
inErrors	The number of inbound packets that contained errors preventing them from being processed correctly.
outOctets	Number of octets sent through the interface.

Physical Interface Counters	Descriptions
ifHCInOctets	The number of octets (8-bit bytes) received by a network interface. It is a part of the SNMP MIB (Management Information Base) structure and is used to track high-capacity input octets counter used in the context of SNMP monitoring.
outUcastPkts	Number of unicast packets sent through the interface.
outNonUcastPkts	Number of non-unicast packets sent through the interface.
outErrors	The number of outbound packets that contained errors preventing them from being processed correctly.
etherStatsDropEvents	Number of events where packets were not delivered to the protocol stack because of resource limitations or other reasons. These dropped events can occur due to buffer overflows, congestion, or hardware limitations.
etherStatsMulticastPkts	The number of packets received by an interface that were addressed to a multicast MAC address.
etherStatsBroadcastPkts	The number of broadcast packets received on an Ethernet interface.
etherStatsUndersizePkts	The number of received packets that are smaller than the minimum allowed Ethernet frame size. Undersized packets can indicate various issues.
etherStatsFragments	The number of received packets that are fragments of IP datagrams.
etherStatsOversizePkts	The number of received packets that exceed the maximum Ethernet frame size.
etherStatsOctets	The total number of octets (bytes) of data transmitted and received on an Ethernet interface. This counter provides a measure of the total amount of data traffic on the interface.

Physical Interface Counters	Descriptions
etherStatsPkts	The number of packets transmitted or received by an Ethernet interface. This counter can provide insights into the traffic load and performance of the interface.
dot1dBasePortMtuExceededDiscards	The number of frames that were discarded at an interface as they exceeded the Maximum Transmission Unit (MTU) of the port. This typically happens when a frame is larger than the maximum size allowed on the interface and cannot be fragmented, so it is dropped.
etherStatsTXNoErrors	The number of Ethernet frames transmitted without any errors through the Ethernet interface. Each time a frame is successfully transmitted without encountering any errors, this counter is incremented.
etherStatsRXNoErrors	The number of Ethernet frames received without any errors through the Ethernet interface. Each time a frame is successfully transmitted without encountering any errors, this counter is incremented.
inMulticastPkts	The number of packets received by the interface that were addressed to a multicast address. These counters provide insights into the amount of multicast traffic being received by the interface.
outBroadcastPkts	The number of packets transmitted by the network interface as broadcast packets. These counters can provide insights into the amount of broadcast traffic generated by the interface.
outMulticastPkts	The number of packets transmitted by the interface as multicast packets. These counters can provide insights into the amount of multicast traffic generated by the interface.

Physical Interface Counters	**Descriptions**
outBroadcastPkts	The number of packets transmitted by the interface as broadcast packets. These counters can provide insights into the amount of broadcast traffic generated by the interface, which can be useful for network troubleshooting and monitoring network performance.
bcmReceivedUndersizePkts	The number of undersized packets received by a Broadcom device. Undersized packets are Ethernet frames that are smaller than the minimum allowed size. This counter provides insights into packet size issues.
bcmTransmittedUndersizePkts	The number of undersized packets sent by a Broadcom device. Undersized packets are Ethernet frames that are smaller than the minimum allowed size. This counter provides insights into packet size issues.
etherTxOversizePkts	The number of packets that exceed the maximum transmission unit (MTU) size allowed on an Ethernet interface.
etherStatsJabbers	The number of jabber frames received by an Ethernet interface. Jabber frames are Ethernet frames that exceed the maximum allowed frame size and contain data that extends beyond the maximum length.
etherStatsCRCAlignErrors	The number of frames received by an Ethernet interface that has a CRC (Cyclic Redundancy Check) error. Also, the frames are not an integral number of octets in length; that is, the frame length is not a multiple of 8 bits. CRC errors occur when the CRC checksum calculated by the receiving interface does not match the CRC checksum transmitted by the sending interface, indicating that the frame may have been corrupted during transmission

Physical Interface Counters	Descriptions
dot3StatsFCSErrors	The number of frames that have a Frame Check Sequence (FCS) error received by an Ethernet interface. The FCS is a field in the Ethernet frame that contains a checksum calculated based on the contents of the frame.
ifHCOutMulticastPkts	The number of outbound multicast packets on a network interface. The ifHCOutMulticastPkts object uses a 64-bit counter, allowing it to accommodate high-speed interfaces without wrapping around as quickly. It provides an accurate count of the outbound multicast packets on the interface.
ifHCOutBroadcastPckts	The ifHCOutBroadcastPkts counter provides the number of outbound broadcast packets on an interface. It is part of the IF-MIB (Interface MIB) and is an extension of the standard ifOutBroadcastPkts counter, providing a 64-bit counter for high-speed interfaces to avoid wrapping around quickly.

Permanent ARP Entry on IFLs

The Permanent ARP Entry functionality is supported on IFLs. This functionality enables devices to proactively resolve ARP and ND.

Path MTU Discovery

Path MTU Discovery (PMTUD) is a technique that is used to determine the Maximum Transmission Unit (MTU) size that can be used along the path from a source to a destination without requiring fragmentation. PMTUD helps to avoid fragmentation by dynamically discovering the smallest MTU in the path.

PMTUD is crucial for optimizing network performance by avoiding fragmentation.

By default, path MTU discovery is enabled in RBFS.

When RBFS receives packets that violate the MTU, meaning their size exceeds the allowed MTU size, it will respond with the following ICMP error message:

ICMPv4 Message Types

The type field specifies the message type sent by the host, providing detailed information about the error condition.

The following tables show the ICMPv4 message types.

Type	Description
3	Destination Unreachable. This message notifies the source host about delivery issues faced while attempting to reach the destination.

Destination Unreachable uses the following code values to further describe the function of the ICMP message being sent.

Code	Description
4	Fragmentation Needed and Don’t-Fragment (DF) was Set. This message shows when a router receives a packet that requires fragmentation while having the DF flag enabled.

ICMPv6 Message Types

The type field identifies the type of the message sent by the host. The type field contains more specific information about the error condition.

The table below lists the ICMPv6 message type.

Type	Description
2	Packet Too Big. A Packet Too Big must be sent by a router when it receives a packet that cannot be forwarded because it is larger than the MTU of the outgoing link.

Code	Description
0	No code

You can modify this behavior by enabling fragmentation. For details on how to enable hostpath fragmentation, refer to the section 'Enabling Hostpath Fragmentation' below.

All outgoing packets are validated against the configured MTU on the egress path.

- If MTU is violated and MTU-profile action is drop, then packets are dropped in hardware.
- If MTU is violated and MTU-profile action is redirect-to-cpu, a 20MB policer is used to protect the CPU-port from overwhelming MTU-violated traffic, and packets are sent to the CPU port.
- When fragmentation is enabled, one of the following operations takes place.

  If the DF (Don't Fragment) bit is not set in the received packet (applicable only to IPv4), the packets are fragmented and sent to the outgoing port.

  If the DF bit is set in a packet, it drops the packet and sends an ICMP error message back to the source host.
- If the fragmentation is disabled, packets are dropped and ICMP error messages are sent to the source host.

For information on configuring the MTU profile, see MTU Profile Configuration.

Note: The minimum MTU supported for IPv4 is 576 bytes, while for both IPv6 and PPPoE, it is 1280 bytes.

IP Fragmentation

If the size of the original IP packet exceeds the MTU of the outgoing interface, the packet must be fragmented.

RBFS supports IP fragmentation on its Q2C platforms.

The MTU profile configuration determines if packets exceeding the MTU are dropped or sent to the CPU for further processing.

The forwarding-options configuration includes a setting to enable hostpath fragmentation for packets redirected to the CPU. If this setting is not configured, the device responds with an ICMP error message to the source host.

Guidelines and Limitations of IP Fragmentation

The following guidelines and limitations apply to IP Fragmentation:

- If a packet exceeds the negotiated subscriber MTU size, the packet is either dropped or fragmented based on the configured MTU profile (applicable only to on the Q2C platforms). The host-path fragmentation is not supported; only an ICMP error message is sent to the source host. For information on MTU profile configuration, see section "MTU Profile Configuration".

You can control fragmentation behavior using the set forwarding-options fragmentation ipv4 state CPU command. For more information about configuring fragmentation, see Enabling Hostpath Fragmentation. * Fragmented packets are reinjected into the egress pipeline and forwarded through the regular QoS path.

- No ICMP error messages are generated for MTU-violated multicast packets.

MTU Profile

The Maximum Transmission Unit (MTU) defines the largest packet size that is allowed to transmit across the network.

In the Q2C Platforms, resource efficiency is achieved by using MTU profiles. Components such as IFPs, IFLs, and L3 interfaces reference these MTU profiles. To optimize MTU resource management, you can configure MTU profiles, which can then be applied to various attachment points. The MTU profiles are attached to the physical (IFP), logical (IFL) and L3 interfaces. RBFS supports the following attachment points for MTU profiles.

Attachment Points: The MTU profiles are attached to the interface entities such as physical (IFP), logical (IFL) and L3 interfaces. RBFS supports the following attachment points for MTU profiles:

- Port-level
- L3 interface level (IPv4 and IPv6)
- PPPoE subscriber level (L2 IFL)
- IPoE subscriber level

MTU Size: A user-configured MTU size can range from 64 to 9216 in RBFS.

For MTU profiles of type 'pppoe', users should provide L3 MTU size (IPv4/IPv6 headers).

MTU Type: An MTU type specifies the attachment point of the MTU profile. The MTU types supported are as follows:

- **physical**: When checking the MTU, the total size of the packet, including headers, is considered.
- **ipv4**: The MTU check is conducted based on IPv4 headers.
- **ipv6**: The MTU check is conducted based on IPv6 headers.
- **ip**: MTU profile of type IP and IPoE.
- **pppoe**: The MTU profile is applied to the PPPoE subscriber interface, where you must specify the L3 MTU size. Using its best-match algorithm, the subscriber management service associates these profiles with PPPoE subscribers.

> - MTU profiles for Layer 3 logical interfaces must be explicitly configured. If not set, no default MTU size is applied for either IPv4 or IPv6 traffic.
> - On the Q2C platform, L3 interfaces can be configured with either an IPv4 MTU profile or an IPv6 MTU profile, but not both. However, by using an MTU profile of type 'ip', you can set a common MTU size that applies to both IPv4 and IPv6 traffic.
> - On the QAX platform, only physical MTU profiles are supported.

MTU Action:

The MTU action specifies the response when an MTU check fails. Currently, RBFS supports only the 'drop' action.

MTU Profile Limitations

The MTU profile has the following limitations. Each hardware unit supports a limited number of MTUs.

MTU Profile Limits by Platform

Q2C Platform:

- Total MTU profiles supported: 8
- L3 MTU profiles (IP/IPv4/IPv6): Up to 3
- PPPoE MTU profiles: Up to 6 (including the default PPPoE profile)

- Physical MTU profiles: Up to 7

QAX Platform:

- Physical MTU profiles: 1 only

Interface Holddown

Overview

Interface Holddown or Interface Dampening is a delay mechanism to prevent network instability from flapping interfaces. Interface flapping occurs when an interface goes up and down repeatedly, leading to excessive route recalculations and needless notifications to other software components.

The Interface Holddown feature is used to prevent network instability by temporarily damping the interface state if it is fluctuating. This mechanism suppresses frequently fluctuating routes and ensures that only stable routes are advertised. It improves overall network performance by effectively utilizing resources such as memory and processing power.

The Interface Holddown feature remains supported when interfaces are grouped in a LAG.

Option to Disable

By default, the Interface Holddown feature is enabled on RBFS. You can configure the ‘hold-down down delay’ and ‘hold-down up delay’ parameter values using the configuration commands. You can also disable the feature using the command.

Benefits of Interface Holddown

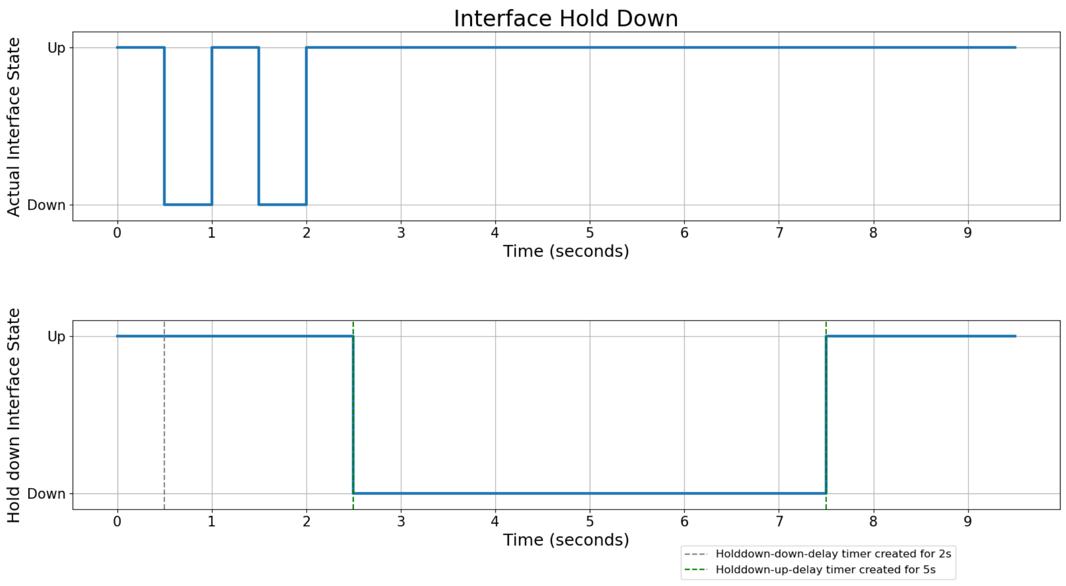
Frequent updates due to route flaps can consume significant processing capacity. This can increase the CPU load on routers, and the constant route advertisements and withdrawals can consume bandwidth. The Interface Holddown mechanism minimizes unnecessary route recalculations and notifications, and the number of updates that need to be processed by suppressing frequently flapping routes. Suppressing unstable routes can improve overall network performance by ensuring that only stable routes are advertised and used for forwarding traffic.

How RBFS Interface Holddown Works

When an interface flaps, such as an interface goes up and down frequently, RBFS employs a holddown period during which the interface's status is retained in its previous state.

The holddown time for an interface increases exponentially within a defined range. The values can be configured explicitly for each interface. When an interface flap occurs, the interface management daemon receives this event from the hardware through the FIBD.

The following diagram shows the interface holddown feature.



The holddown intervals are set at 0ms, 100ms, 200ms, 500ms, 1s, 2s, 5s, and 10s. When the holddown timer expires, the interface state is propagated to higher layers. If the interface remains stable throughout the holddown interval, the actual state is propagated. However, if the interface experiences instability during this interval, it remains in the DOWN state.

RBFS implements "Holddown Down Delay" and "Holddown Up Delay", two parameters that control how the interface flap is managed.

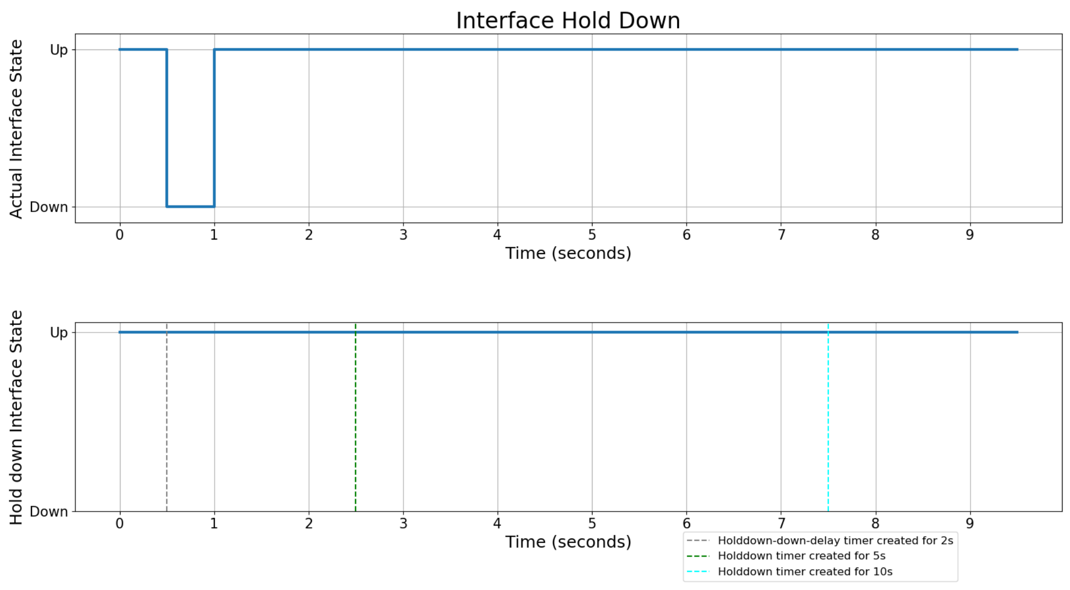
Holddown Down Delay

The 'Hold-down Down Delay' parameter, once configured, ensures that the interface state change from UP to DOWN is delayed by the specified 'holddown down delay' period. After this delay, the default holddown mechanism takes over.

Specifically:

1. The ‘holddown’ interval starts with its initial value, that is 'holddown down delay' period, and increments exponentially when each timer expires.
2. DOWN to UP state changes are delayed based on exponentially increasing intervals.

The following diagram shows 'holddown down delay' scenario.



For example, if the ‘holddown down delay’ is set to 5 seconds and the interface state changes from UP to DOWN, the interface remains up for 5 seconds. After this 5-second delay, the system checks the stability of the interface:

- If the interface has no flap during this period, it remains up.
- If the interface has flapped during this period, it changes to the DOWN state after the delay.

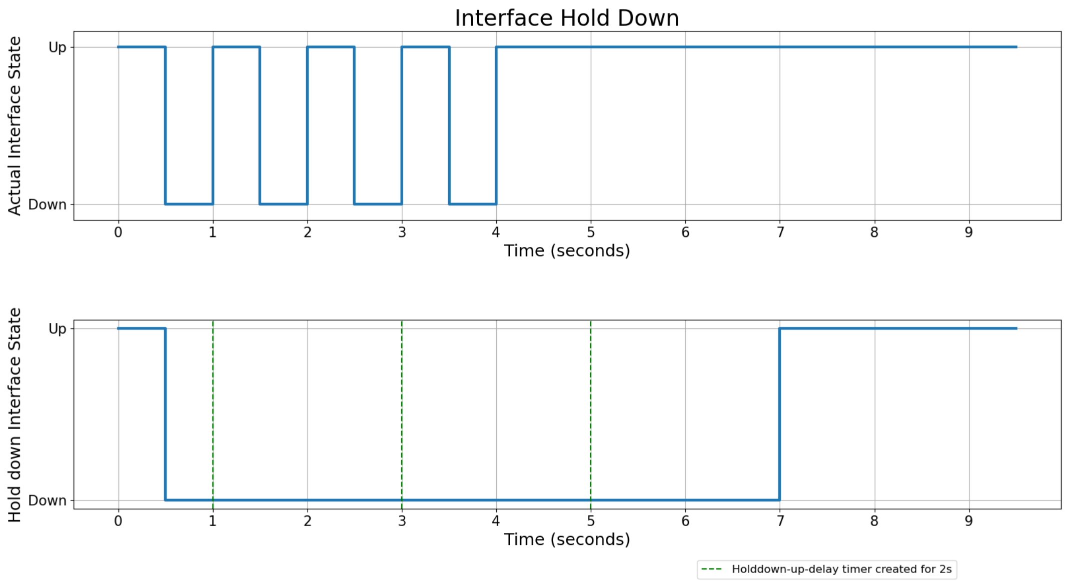
In summary, Holddown Down Delay ensures an interface remains up for a specified time before propagating any state change to higher-level software components.

Holddown Up Delay

'Holddown Up Delay' is the period that an interface stays down after a failure before it is considered to be in a stable state. When the state changes from DOWN to UP, a holddown timer is initiated with a specified delay period. If there are flaps during the previous interval time, then a new timer with the same 'holddown up' delay is set.

This results in a delay in the state transitioning to UP based on the ‘holddown up delay’. However, if state changes happen within the 'holddown up delay' interval, any change from UP to DOWN is immediately transmitted. It uses a static timer instead of an exponential backoff for the holddown interval.

The following diagram shows 'holddown up' scenario.



Example Scenario

Imagine where a router’s interface occasionally loses and regains connectivity due to a loose cable. Each time this happens, it creates an event that propagates through many of the software components in the network.

Without the Interface Holddown feature, every time disruption immediately triggers network protocols and higher-level software components to respond, potentially causing routing updates, recalculations, and additional processing overhead. This constant back and forth leads to network instability and adds additional strain on network devices.

With the interface Holddown enabled, the RBFS still receives these flap events but employs a hold-down mechanism to delay taking action on these changes. For example, with the holddown intervals: 0ms, 100ms, 200ms, 500ms, 1s, 2s, 5s, and 10s, the system performs the following:

- Retain the old interface status when an interface flap is detected.
- Wait for the initial hold-down time, which is 100ms.
- If another transition occurs within this period, the hold-down time increases to the next interval that is, 200ms. This process repeats, increasing the hold-down time after each transition up to the configured maximum value, which is 10s.

In this way, by delaying the processing of interface flaps, interface dampening reduces the impact of flapping on the network. The system only updates the interface status after a stable period without transitions, thus preventing frequent, unnecessary updates.

IPv6 Link-Local Address

In IPv6 networking, link-local addresses are essential for local communication between nodes on the same link. It facilitates communication between nodes on the same link, however cannot be used to send packets beyond that link.

In RBFS, a link-local IPv6 address can be either automatically generated or manually assigned.

When a configured link-local address is deleted, a unique link-local address is automatically generated for the logical interface. This automatically assigned address is typically derived from the interface’s MAC address and other parameters.

You can also modify an existing automatically generated link-local address by manually configuring a link-local IPv6 address on a logical interface. It will replace the auto-generated link-local address with the one you have configured.

Guidelines & Limitations

QAX-based Platforms

An additional restriction applies to ports on QAX-based platforms: because of hardware design, physical ports are grouped into quads (groups of 4, also known as port groups). Each quad must have the same physical parameters: speed, link-training, duplex.

The following tables are provided for easy identification of ports that need to have the same physical settings:

Edgecore 7316-26XB Port Groups:

Port	Speed	Duplex	Port Group
ifp-0/0/0	100G	Full	0
ifp-0/0/1	100G	Full	1
ifp-0/1/0	10G	Full	2
ifp-0/1/1	10G	Full	
ifp-0/1/2	10G	Full	
ifp-0/1/3	10G	Full	
ifp-0/1/4	10G	Full	3
ifp-0/1/5	10G	Full	
ifp-0/1/6	10G	Full	
ifp-0/1/7	10G	Full	
ifp-0/1/8	10G	Full	4
ifp-0/1/9	10G	Full	
ifp-0/1/10	10G	Full	
ifp-0/1/11	10G	Full	
ifp-0/1/12	10G	Full	5
ifp-0/1/13	10G	Full	
ifp-0/1/14	10G	Full	
ifp-0/1/15	10G	Full	
ifp-0/1/16	25G	Full	6
ifp-0/1/17	25G	Full	
ifp-0/1/18	25G	Full	
ifp-0/1/19	25G	Full	

Port	Speed	Duplex	Port Group
ifp-0/1/20	25G	Full	7
ifp-0/1/21	25G	Full	
ifp-0/1/22	25G	Full	
ifp-0/1/23	25G	Full	

UfiSpace S9500-22XST Port Groups:

Port	Speed	Duplex	Port Group
ifp-0/0/0	10G	Full	8
ifp-0/0/1	10G	Full	
ifp-0/0/2	10G	Full	
ifp-0/0/3	10G	Full	
ifp-0/0/4	10G	Full	4
ifp-0/0/5	10G	Full	
ifp-0/0/6	10G	Full	
ifp-0/0/7	10G	Full	
ifp-0/0/8	10G	Full	11
ifp-0/0/9	10G	Full	
ifp-0/0/10	10G	Full	
ifp-0/0/11	10G	Full	
ifp-0/0/12	25G	Full	3
ifp-0/0/13	25G	Full	
ifp-0/0/14	25G	Full	
ifp-0/0/15	25G	Full	
ifp-0/0/16	25G	Full	2
ifp-0/0/17	25G	Full	
ifp-0/0/18	25G	Full	
ifp-0/0/19	25G	Full	

Port	Speed	Duplex	Port Group
ifp-0/0/20	100G	Full	0
ifp-0/0/21	100G	Full	1

A PHY Quad can be associated with network interface (NIF) ports of identical type only. For example, a quad cannot be a mix of XLGE and XE ports. An exception is GE and XE ports which can coexist in the same quad. This means all the ports in a port group should have the same physical interface configuration (that is, speed/duplex/link-training). Ports in a port group are only allowed to support 1G and 10G speeds; any other combination is not allowed. If a port within a port group is misconfigured, then it would require changing the speeds/interface type of all ports within the port group to a different type and then back into the original type.

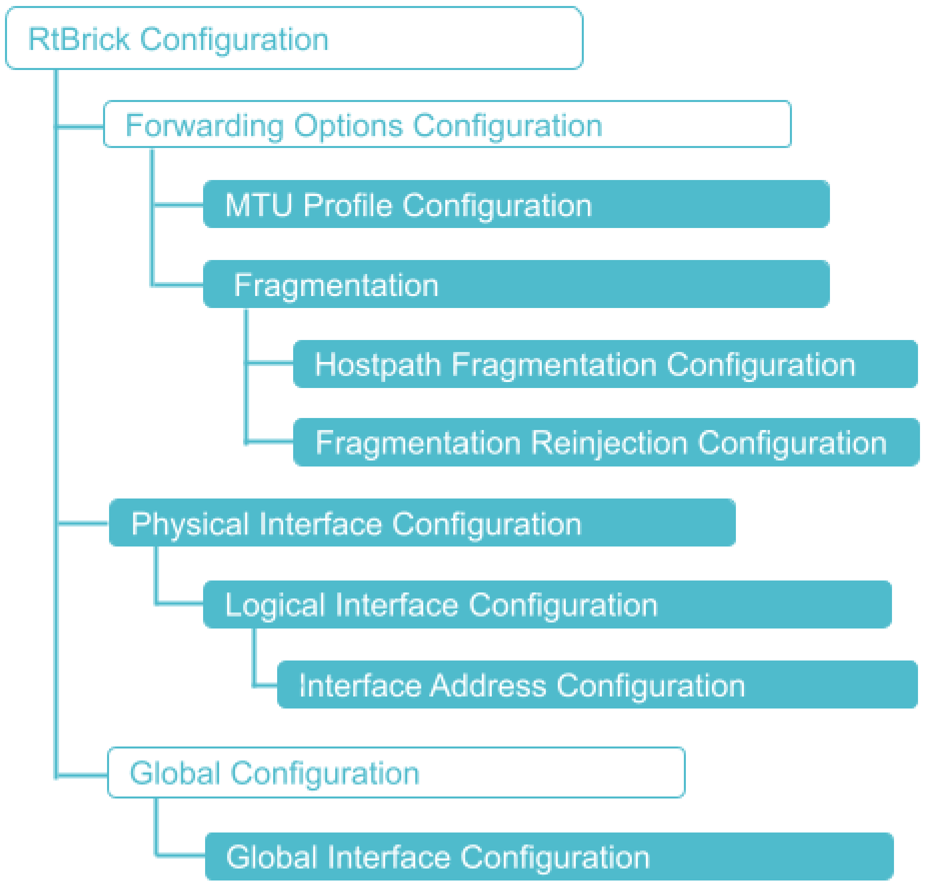
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the Platform Guide for the features and the sub-features that are or are not supported by each platform.

6.1.2. Interfaces Configuration

Configuration Hierarchy

The diagram illustrates the interface configuration hierarchy.


RtBrick Configuration
Forwarding Options Configuration
MTU Profile Configuration
Fragmentation
Hostpath Fragmentation Configuration
Fragmentation Reinjection Configuration
Physical Interface Configuration
Logical Interface Configuration
Interface Address Configuration
Global Configuration
Global Interface Configuration

Configuration Syntax and Commands

The following sections describe the interface configuration syntax and commands.

MTU Profile Configuration

This section describes how to configure MTU profiles.

Syntax:

set forwarding-options mtu-profile <attribute> <value>

Attribute	Description
mtu-profile <mtu-profile-name>	MTU profile name
mtu-size <mtu-size>	MTU size. Range: 64 to 9216 bytes

Attribute	Description
mtu-type <mtu-type>	Specify the MTU type: • **physical**: Port based MTU profile • **pppoe**: subscriber IFL-based MTU profile for L2TP and PPPoE. This MTU profile is used by PPPoE subscribers to set the default MTU size of 1492. A configured size of 1492 bytes limits the size of the IPv4 or IPv6 header plus payload. • **ipv4**: MTU profile of type IPv4. Only IPv4 traffic on the logical interface will be impacted. • **ipv6**: MTU profile of type IPv6. Only IPv6 traffic on the logical interface will be impacted. • **ip**: MTU profile of type IP. Both IPv4 and IPv6 traffic on the logical interface will be impacted.
action <mtu-action>	Specify the MTU action. The following options are supported: **drop**: This indicates that when the MTU check fails, the action "drop" is performed. **redirect-to-cpu**: This is an action of redirecting packets to the CPU in a traffic behavior. A redirect-to-cpu action must be configured for fragmentation to occur.

Example 1: Configuration of the MTU Profile for the Physical Port

```
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "mtu-profile": [
        {
          "mtu-profile-name": "portMtu",
          "size": 5000,
          "type": "physical",
          "action": "redirect-to-cpu"
        }
      ]
    }
  }
}
```

Example 2: MTU Profile Configuration of Type IPv4

```
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "mtu-profile": [
        {
          "mtu-profile-name": "ipv4Mtu",
          "size": 1300,
          "type": "ipv4",
          "action": "redirect-to-cpu"
        }
      ]
    }
  }
}
```

Example 3: MTU Profile Configuration of Type IPv6

```
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "mtu-profile": [
        {
          "mtu-profile-name": "ipv6Mtu",
          "size": 1400,
          "type": "ipv6",
          "action": "redirect-to-cpu"
        }
      ]
    }
  }
}
```

Example 4: Configuration of the MTU Profile for PPPoE

```
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "mtu-profile": [
        {
          "mtu-profile-name": "pppoeMtu",
          "size": 1492,
          "type": "pppoe",
          "action": "redirect-to-cpu"
        }
      ]
    }
  }
}
```

Enabling Hostpath Fragmentation

This section describes how to enable or disable fragmentation by CPU. It is necessary to configure MTU profile action "redirect-to-cpu" so that fragmentation takes place. By default, fragmentation is disabled.

Syntax:

set forwarding-options fragmentation ipv4 state <value>

Attribute	Description
disabled | cpu	Enables fragmentation of IPv4 packets. There are two options: disabled—Fragmentation is disabled. It is the default setting. cpu—Fragmentation is performed by CPU.

Example: Configuration of Hostpath Fragmentation

```
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "fragmentation": {
        "ipv4": {
          "state": "cpu"
        }
      }
    }
  }
}
```

Configuring Policer Limits for Fragmented Traffic

RBFS optimizes performance for fragmented traffic by adjusting the default policer limits on the host path. Fragmentation packets received on the host path has a cap upto 150 Mbps, which enables higher throughput for fragmented traffic

Syntax:

set forwarding-options fragmentation ipv4 state <value>

set forwarding-options fragmentation ipv4 policer <policer-name>

The following policer configuration sets the RBFS host path to support 120 Mbps.

```
set forwarding-options fragmentation ipv4 state cpu
set forwarding-options fragmentation ipv4 policer policer_120MB
```

Example:

```
supervisor@rtbrick>: cfg> show config forwarding-options fragmentation
{
  "rtbrick-config:fragmentation": {
    "ipv4": {
    "state": "cpu",
    "policer": "policer_120MB"
    }
  }
}
```

- RBFS supports default policers, which can be applied to fragmentation traffic through the policer configuration.
- In addition to the default policer, users can define a custom policer and attach it to the fragmentation configuration. However, the committed information rate (CIR) cannot exceed 150Mbps.

Physical Interface Configuration

This section describes configuration options at the physical interface (IFP) level.

Syntax:

set interface <interface-name> <attribute> <value>

Attribute	Description
<interface-name>	Name of the interface. Example: ifp-0/0/1.
admin-status <down|up>	Administrative state of the interface.
auto-negotiation true	Enable auto-negotiation. To disable auto-negotiation, use the delete form of the command. Port speed configuration and auto-negotiation are mutually exclusive.

Attribute	Description
holddown <disable|enable>	Enable or disable the holddown timer feature. By default, the feature is enabled.
holddown-down-delay <...>	Configure the down delay time in milliseconds (ms).
holddown-up-delay <...>	Configure the down delay time in milliseconds (ms).
class-of-service <profile-name>	Apply class-of-service profile name.
description	Configure physical interface description.
host-if <container-interface>	Configure a host interface. For example, if the container interface eth1 connects to the host interface vethXYZ123, use this command option to bound hostif-0/0/1 to eth1. The Linux virtual ethernet (veth) interface needs to be created separately. It cannot be created via RBFS configuration.
forward-error-correction <fec-type>	Configure Forward Error Correction (FEC) on the physical interface. FEC allows you to send the required information to correct errors through the link along with the payload data. A benefit of "forward" in FEC is that errors detected at the receiver do not need to be retransmitted. Currently, the supported FEC types are: base-r, rsfec, none. NOTE: rsfec is the only FEC supported for 100G on the QAX platform.
link-training true	Enable link training. To disable link training, use the delete form of the command.

Attribute	Description
master <true|false>	Memif role, master or slave, applicable only to memif interface. One end needs to be configured as master, and the other one as slave.
memif-id <id>	Configure memif ID, applicable only to memif interface. Needs to match on both ends.
mtu-profile <mtu-profile-name>	Attach MTU profile to a physical interface. This is a mandatory attribute.
mru <size>	Maximum receive unit size on the physical interface.
speed <speed>	Configure speed mode for the interface. Port speed refers to the maximum amount of data transmitted. The speed value is specified in Gigabits per second (Gbps). Currently, RBFS supports 10G and 100G ports, and you can make the following changes: • 100G port speed can be changed to 40G • 10G port speed can be changed to 1G

Example 1: Physical Interface Configuration

```
{
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/1",
        "description": "Link to leaf1",
        "speed": "10G",
        "mtu-profile": "portMtu",
        "mru": 5000
      }
    ]
}
```

Example 2: Memory Interface Configuration

A End:

```
{
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/1",
        "description": "Master",
        "memif-id": 11,
        "master": "true",
      }
    ]
}
```

B End:

```
{
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/1",
        "description": "Slave",
        "memif-id": 11,
        "master": "false",
      }
    ]
  }
```

Example 3: Host Interface Configuration

```
{
    "rtbrick-config:interface": [
        "name": "ifp-0/0/1",
        "description": "Represents eth1 as ifp-0/0/1 in RBFS",
        "host-if": "eth1",
    ]
}
```

Example 4: MRU Configuration for Physical Interface

```
{
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/7",
        "mru": 5000
      }
    ]
  }
```

Example 5: FEC Configuration for Physical Interface

```
{
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/40",
        "forward-error-correction": "base-r"
        }
      ]
    }
```

To access the Operational State API that corresponds to this CLI, click here.

Logical Interface Configuration

This section describes configuration options at the logical interface (IFL) level.

Syntax:

set interface <interface-name> **unit** <unit-id> <attribute> <value>

Attribute	Description
unit <unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface.
admin-status <down|up>	Administrative state of the logical interface.
class-of-service <profile-name>	Apply class-of-service profile name.
description <description>	Description of the logical interface.
inner-vlan <inner-vlan-id>	Inner VLAN ID.
instance <instance>	Assign the logical interface to an instance.
ipv4-admin-status <down|up>	Enable or disable IPv4.
ipv4-mtu-profile <ipv4-mtu-profile>	Attach IPv4 MTU profile to an L3 interface.
ipv6-admin-status <down|up>	Enable or disable IPv6.
ipv6-mtu-profile <ipv6-mtu-profile>	Attach IPv6 MTU profile to an L3 interface.
ip-mtu-profile <ip-mtu-profile>	Attach IP MTU profile to an L3 interface.

Attribute	Description
mpls-admin-status <down | up>	Enable or disable MPLS.
mpls-mtu <mpls-mtu-size>	MPLS maximum transmission unit size.
neighbor <ipv4 | ipv6> <ip-address> mac <mac-address>	Configure a static IPv4 or IPv6 neighbor.
unnumbered interface <loopback-interface-name>	Configure an un-numbered interface.
vlan <outer-vlan-id>	Outer VLAN ID.
vlan-encapsulation <802.1ad | 802.1q>	Specify the outer VLAN tag.

Example 1: Logical Interface Configuration with IPv4 MTU Profile

```
{
  "rtbrick-config:interface": [
     {
       "name": "ifp-0/0/1",
       "unit": [
         {
           "unit-id": 1,
           "description": "VLAN 101",
           "instance": "default",
           "ipv4-mtu-profile": "ipv4Mtu"

         }
       ]
     }
  ]
}
```

Example 2: Logical Interface Configuration with IPv6 MTU Profile

```
{
  "rtbrick-config:interface": [
     {
       "name": "ifp-0/0/1",
       "unit": [
         {
           "unit-id": 1,
           "description": "VLAN 101",
           "instance": "default",
           "ipv6-mtu-profile": "ipv6Mtu"
```

```
          }
        ]
      }
   ]
}
```

Example 3: Logical Interface Configuration with IP MTU Profile

```
 {
   "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/1",
        "unit": [
          {
            "unit-id": 1,
            "description": "VLAN 101",
            "instance": "default",
            "ip-mtu-profile": "ipMtu"

          }
        ]
      }
   ]
}
```

Interface Address Configuration

This section describes how to configure interface IP addresses.

Syntax:

set interface <interface-name> **unit** <unit-id> **address** <afi> <attribute> <value>

Attribute	Description
<afi>	Address family identifier (AFI). Supported values: ipv4 and ipv6
<prefix4|prefix6>	Assign IPv4 or IPv6 address to the interface unit.
community <community-value>	Configure list of communities associated with the address.
extended-community <community-value>	Configure list of extended communities associated with the address.

Attribute	Description
label <label-value>	Configure label associated with the address. Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. It is recommended to assign label values outside of these reserved ranges to avoid conflicts.
secondary true	Enable a secondary IPv4 address. To disable the secondary IP configuration, use the delete form of the command.

Broadcast and network IP addresses cannot be configured on logical (IFL) interfaces.

Example: Interface Address Configuration

```
{
    "rtbrick-config:interface": [
      {
        "name": "lo-0/0/1",
        "unit": [
          {
            "unit-id": 1,
            "address": {
              "ipv4": [
                {
                  "prefix4": "198.51.100.103/24",
                  "label": 12346
                }
              ]
            }
          }
        ]
      }
    ]
  }
```

Global Interface Configuration

This section describes a configuration option applied globally to all interfaces.

Syntax:

set global interface all <attribute> <value>

Attribute	Description
admin-status <up|down>	Configure state of the interface.

- The interface level enable/disable command has higher precedence than the global interface enable/disable command.
- You can disable all unused physical interfaces.
- Before executing the global interface disable all command ensure that all physical interfaces are in the link Up state.

Example: Enabling or Disabling all Interfaces

```
{
  "ietf-restconf:data": {
    "rtbrick-config:global": {
      "interface": {
        "all": {
          "admin-status": "down"
        }
      }
    }
  }
}
```

Interface Holddown Configuration

By default, the Interface Hold-down feature is enabled. You can disable the feature if you want. It is allowed to configure timer value for options such as 'holddown down delay' and 'holddown up delay'.

Syntax:

set interface <interface-name> <attribute> <value>

Attribute	Description
<interface-name>	Name of the interface. Example: ifp-0/0/1.
holddown <disable|enable>	Enable or disable the holddown timer feature. By default, the feature is enabled.

Attribute	Description
holddown-down-delay	Configure the down delay time in milliseconds (ms). The allowed values include 0, 100, 200, 500, 1000, 2000, 5000, and 10000. It overrides the default delay of '0' milliseconds.
holddown-up-delay	Configure the up delay time milliseconds (ms). The allowed values include 0, 100, 200, 500, 1000, 2000, 5000, and 10000.

IPv6 Link-Local Address Configuration

You can manually assign an IPv6 link-local address on an interface.

Syntax:

set interface <interface-name> **unit** <unit-id> **address ipv6-link-local** <IPv6 address>

Attribute	Description
<interface-name>	Name of the interface. Example: ifp-0/0/1.
unit <unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface.
ipv6-link-local <IPv6 address>	Specifies the IPv6 link-local address.

Example for IPv6 link-local configuration

```
{
 "ietf-restconf:data": {
   "rtbrick-config:interface": [
     {
       "name": "hostif-0/0/0",
       "host-if": "B1-1-SN"
     },
     {
       "name": "hostif-0/0/2",
       "host-if": "B1-3-B2",
       "unit": [
         {
           "unit-id": 10,
           "address": {
             "ipv6-link-local": "fe80::1"
           }
         }
       ]
     },
     {
       "name": "hostif-0/0/3",
       "host-if": "B1-4-B2"
     }
   ]
 }
}
```

Enabling Gratuitous ARP on Interface

You can enable or disable the sending of Gratuitous ARP (GARP) on an interface. This section describes enabling or disabling GARP on an interface.

Syntax:

set interface <interface-name> **unit** <unit-id> **gratitous-arp** <enable|disable>

Attribute	Description
interface <interface-name>	Name of the interface. Examples: ifp-0/0/1.
gratitous-arp	Specify enable or disable.

Example: Enabling Gratitous ARP

```
supervisor@rtbrick.net: cfg> set interface hostif-0/0/0 unit 100 gratitous-arp
enable|disable
```

Permanent ARP Entry Configuration on Logical Interfaces

This Permanent ARP Entry on IFLs enables the RBFS devices to proactively resolve ARP and ND.

Syntax:

set interface <interface> unit <unit id> **neighbor** < **ipv4** | **ipv6** > <ip address> **mac dynamic true**

The following example shows how to configure an ARP entry on a logical interface.

```
set interface ifp-0/1/0 unit 6 neighbor ipv4 123.1.1.1 mac dynamic true
```

Example:

```
{
  "ietf-restconf:data": {
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/0",
        "host-if": "S1-1-SN",
        "unit": [
          {
            "unit-id": 10,
            "neighbor": {
              "ipv4": [
                {
                  "address4": "10.2.2.2",
                  "dynamic": "true"
                }
              ]
            }
          }
        ]
      },
      {
        "name": "ifp-0/0/1",
        "host-if": "S1-2-SN"
      },
      {
        "name": "ifp-0/0/2",
        "host-if": "S1-3-L1"
      },
      {
        "name": "ifp-0/0/3",
        "host-if": "S1-4-L1"
      }
    ]
  }
}
```

Breakout Interfaces

Using the breakout configuration, the high-speed interfaces(100G) on the UfiSpace S9600-32X platform can be divided into multiple lower-speed interfaces(10G or 25G), allowing them to connect with lower rate ports(10G or 25G) in other network devices.

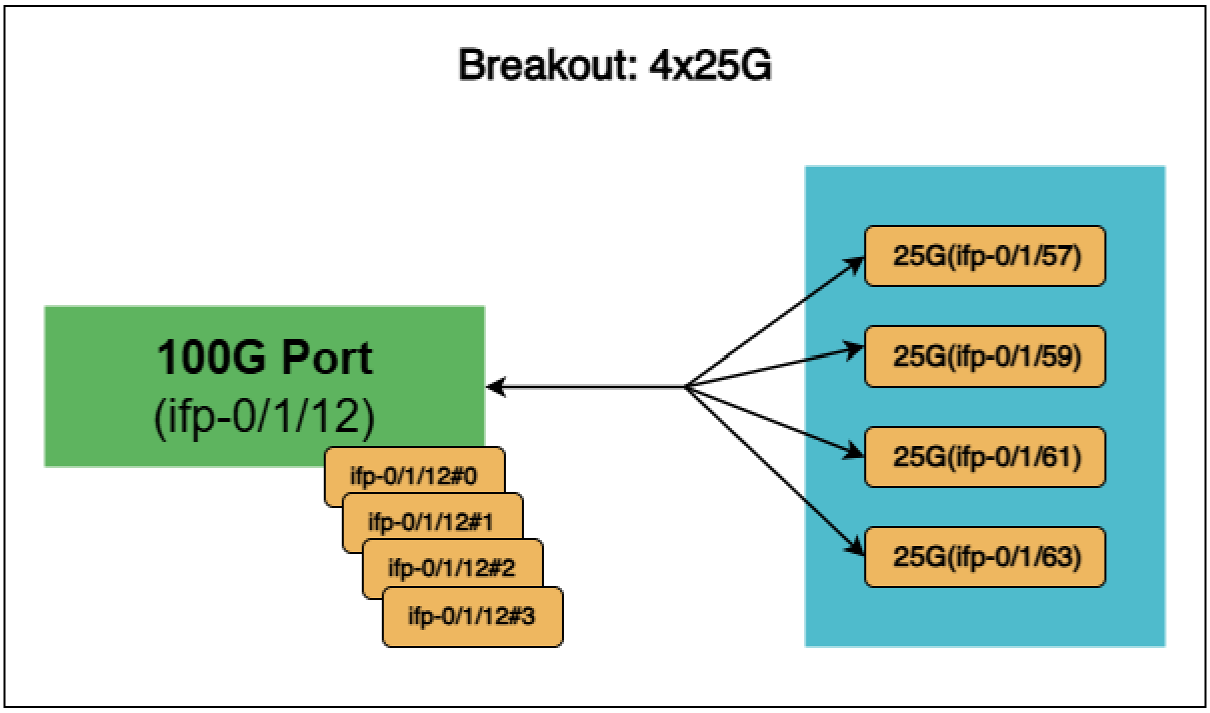
RBFS supports the following breakout configuration options:

Rate	Technology	Breakout Capable	Electric Lanes	Optical Lanes
100G	QSFP28	Yes	4 x 25G	4 x 25G
100G	QSFP28	Yes	4 x 10G	4 x 10G

For example, to set up the individual peer-side (25G/10G) interfaces, we need to configure the breakout on the 100G port to 4x10G or 4x25G based on the peer-side speed.

When a breakout is configured on a high-speed interface (for example, ifp-0/1/12), it is divided into four lower-speed interfaces: ifp-0/1/12#0, ifp-0/1/12#1, ifp-0/1/12#2, and ifp-0/1/12#3. As a result, the original high-speed interface will no longer be available.

The following diagram shows the breakout for the interface ifp-0/1/12 on the high-speed 100G to 4x25G ports (#0, #1, #2, #3).



Guidelines and Limitations

Currently, the following limitations exist.

- Autonegotiation, speed configuration, and traffic sampling are not supported for breakout interfaces.
- A breakout interface cannot be a member of a LAG.
- Using breakout interfaces with routing protocols is not supported.

Configuring Breakout

To configure for the interface ifp-0/1/12 on the high-speed 100G to 4x25G ports., enter the following command:

Syntax:

set interface <interface-name> **breakout map** < **4x10G** | **4x25G** >

Example:

```
set interface ifp-0/1/12 breakout map 4x25G
```

Viewing Breakout Configuration Details

To view the breakout configuration details, enter the following command:

```
show interface <interface-name> detail
```

Example of Breakout Configuration:

```
supervisor@rtbrick>SPINE01: cfg> show interface ifp-0/1/12#0 detail
Interface: ifp-0/1/12#0
  Admin/Link/Operational status: Up/Up/Up
  Damping Status: holddown
  Breakout: 4x25G
  Speed configured: 25G
  Speed maximum: 25G
  Duplex: Full
  Encapsulation mode: ieee
  MRU: 10000
  MTU: 9000
  Maximum frame size: 10000
  Interface type: ethernet
  Interface index: 100609
  MAC: e8:c5:7a:8f:56:c6
  Uptime:  Mon Nov 25 05:37:10 GMT +0000 2024, Holddown up/down delay: 10000ms/0ms
  Flap count:  1
  Holddown transitions:  1
  Description: Physical interface #12 from node 0, chip 1
  LED Status: Off, LED programmed value: 82
  Packet statistics:
    Rx packets: 3          Tx packets: 3
    Rx bytes: 381          Tx bytes: 387
```

```
supervisor@rtbrick>SPINE01: cfg> show interface ifp-0/1
/12#1 detail
Interface: ifp-0/1/12#1
  Admin/Link/Operational status: Up/Up/Up
  Damping Status: stable
  Breakout: 4x25G
  Speed configured: 25G
  Speed maximum: 25G
  Duplex: Full
  Encapsulation mode: ieee
  MRU: 10000
  MTU: 9000
  Maximum frame size: 10000
  Interface type: ethernet
  Interface index: 102657
  MAC: e8:c5:7a:8f:56:c7
  Uptime:  Mon Nov 25 05:37:10 GMT +0000 2024, Holddown up/down delay: 10000ms/0ms
  Flap count:  1
  Holddown transitions:  1
  Description: Physical interface #12 from node 0, chip 1
  LED Status: Off, LED programmed value: 4
  Packet statistics:
    Rx packets: 1          Tx packets: 1
    Rx bytes: 127          Tx bytes: 129
```

```
supervisor@rtbrick>SPINE01: cfg> show interface ifp-0/1
/12#2 detail
Interface: ifp-0/1/12#2
  Admin/Link/Operational status: Up/Up/Up
  Damping Status: stable
  Breakout: 4x25G
  Speed configured: 25G
  Speed maximum: 25G
  Duplex: Full
  Encapsulation mode: ieee
  MRU: 10000
  MTU: 9000
  Maximum frame size: 10000
  Interface type: ethernet
  Interface index: 104705
  MAC: e8:c5:7a:8f:56:c8
  Uptime:  Mon Nov 25 05:37:10 GMT +0000 2024, Holddown up/down delay: 10000ms/0ms
  Flap count:  1
  Holddown transitions:  1
  Description: Physical interface #12 from node 0, chip 1
  LED Status: Off, LED programmed value: 0
  Packet statistics:
    Rx packets: 1          Tx packets: 1
    Rx bytes: 127          Tx bytes: 129
```

```
supervisor@rtbrick>SPINE01: cfg> show interface ifp-0/1
/12#3 detail
Interface: ifp-0/1/12#3
  Admin/Link/Operational status: Up/Up/Up
  Damping Status: stable
  Breakout: 4x25G
  Speed configured: 25G
  Speed maximum: 25G
  Duplex: Full
  Encapsulation mode: ieee
  MRU: 10000
  MTU: 9000
  Maximum frame size: 10000
  Interface type: ethernet
  Interface index: 106753
  MAC: e8:c5:7a:8f:56:c9
  Uptime:  Mon Nov 25 05:37:10 GMT +0000 2024, Holddown up/down delay: 10000ms/0ms
  Flap count:  1
  Holddown transitions:  1
  Description: Physical interface #12 from node 0, chip 1
  LED Status: Off, LED programmed value: 0
  Packet statistics:
    Rx packets: 1          Tx packets: 1
    Rx bytes: 127          Tx bytes: 129
```

6.1.3. Interfaces Operational Commands

Interface Show Commands

The interface show commands provide detailed information about the status and parameters of RBFS interfaces.

Interface Summary Commands

Syntax:

show interface <option>

Option	Description
summary	Displays a summary of all interfaces including physical, logical, and address information.
<interface-name>	Displays a summary of an interface including physical, logical, and address information.
physical	Displays all physical interface including loopback, cpu and recycle ports.
logical	Displays all logical interfaces for all instances.
logical <instance-name>	Displays all logical interfaces for the given instance.
address	Displays all IPv4 and IPv6 addresses for all instances.
address <instance-name>	Displays all IPv4 and IPv6 addresses for the given instance.

Example 1: Summary Output for All Interfaces

```
supervisor@rtbrick>LEAF01: op> show interface summary
Interface                   Admin     Link      Oper           IPv4 Address              IPv6 Address
ifp-0/0/1                   Up        Down      Down
ifp-0/0/2                   Up        Down      Down
ifp-0/0/3                   Up        Down      Down
ifp-0/0/4                   Up        Up        Up
ifp-0/0/5                   Up        Down      Down
```

```
ifp-0/0/6                Up       Down     Down
ifp-0/0/7                Up       Down     Down
ifp-0/0/8                Up       Down     Down
ifp-0/0/9                Up       Down     Down
ifp-0/0/10               Up       Up       Up
ifl-0/0/10/100           Up       Up       Up          198.51.100.22/24         2001:db8:0:100::/32
ifl-0/0/10/200           Up       Up       Up          198.51.100.32/24         2001:db8:0:10::/32
ifl-0/0/10/300           Up       Up       Up          -                        2001:db8:0:160::/32
ifp-0/0/11               Up       Down     Down
ifp-0/0/12               Up       Down     Down
ifp-0/0/13               Up       Down     Down
ifp-0/0/14               Up       Down     Down
ifp-0/0/15               Up       Down     Down
ifp-0/0/16               Up       Down     Down
ifp-0/0/17               Up       Down     Down
ifp-0/0/18               Up       Down     Down
ifp-0/0/19               Up       Down     Down
ifp-0/0/20               Up       Down     Down
ifp-0/0/21               Up       Down     Down
ifp-0/0/22               Up       Down     Down
ifp-0/0/23               Up       Down     Down
ifp-0/0/24               Up       Down     Down
ifp-0/0/25               Up       Down     Down
ifp-0/0/26               Up       Down     Down
ifp-0/0/27               Up       Up       Up
ifp-0/0/28               Up       Down     Down
ifp-0/0/29               Up       Down     Down
ifp-0/0/30               Up       Down     Down
ifp-0/0/31               Up       Down     Down
ifp-0/0/32               Up       Down     Down
ifp-0/0/33               Up       Down     Down
ifp-0/0/34               Up       Down     Down
ifp-0/0/35               Up       Down     Down
ifp-0/0/36               Up       Down     Down
ifp-0/0/37               Up       Down     Down
ifp-0/0/38               Up       Down     Down
ifp-0/0/39               Up       Down     Down
ifp-0/0/40               Up       Down     Down
ifp-0/0/41               Up       Down     Down
ifp-0/0/42               Up       Down     Down
ifp-0/0/43               Up       Down     Down
ifp-0/0/44               Up       Down     Down
ifp-0/0/45               Up       Down     Down
ifp-0/0/46               Up       Down     Down
ifp-0/0/47               Up       Down     Down
ifp-0/0/48               Up       Down     Down
ifp-0/0/49               Up       Down     Down
ifp-0/0/50               Up       Down     Down
ifp-0/0/51               Up       Down     Down
ifp-0/0/52               Up       Up       Up
ifp-0/0/53               Up       Up       Up
ifp-0/0/54               Up       Down     Down
cpu-0/0/200              Up       Up       Up
cpu-0/0/201              Up       Down     Down
cpu-0/0/202              Up       Down     Down
cpu-0/0/203              Up       Down     Down
recycle-0/0/75           Up       Up       Up
recycle-0/0/75/0         Up       Up       Up
recycle-0/0/76           Up       Up       Up
recycle-0/0/76/0         Up       Up       Up
```

Example 2: Summary Output for One Physical Interface

```
supervisor@rtbrick>LEAF01: op> show interface ifp-0/0/10
Interface                Admin    Link     Oper          IPv4 Address             IPv6 Address
ifp-0/0/10               Up       Up       Up
  ifl-0/0/10/100         Up       Up       Up            198.51.100.22/24               2001:db8:0:100::/32
  ifl-0/0/10/200         Up       Up       Up            198.51.100.32/24               2001:db8:0:10::/32
  ifl-0/0/10/300         Up       Up       Up            -                        2001:db8:0:160::/32
  ifl-0/0/10/1000        Up       Up       Up           -                      2001:db8:0:33::/32
```

Example 3: List of All Physical Interfaces

```
supervisor@rtbrick>LEAF01: op> show interface physical
Interface        Admin   Link    Oper    MAC Address         Speed   Duplex  Uptime
lo-0/0/1         Up      Up      Up      80:a2:35:a0:00:01   -       -       Thu Nov 19 10:41:06 GMT +0000
2020
ifp-0/0/1        Up      Down    Down    80:a2:35:ee:a8:01   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/2        Up      Down    Down    80:a2:35:ee:a8:02   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/3        Up      Down    Down    80:a2:35:ee:a8:03   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/4        Up      Up      Up      80:a2:35:ee:a8:04   10G     Full    Thu Nov 19 10:05:02 GMT +0000
2020
ifp-0/0/5        Up      Down    Down    80:a2:35:ee:a8:05   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/6        Up      Down    Down    80:a2:35:ee:a8:06   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/7        Up      Down    Down    80:a2:35:ee:a8:07   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/8        Up      Down    Down    80:a2:35:ee:a8:08   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/9        Up      Down    Down    80:a2:35:ee:a8:09   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/10       Up      Up      Up      80:a2:35:ee:a8:0a   10G     Full    Fri Nov 20 00:59:12 GMT +0000
2020
ifp-0/0/11       Up      Down    Down    80:a2:35:ee:a8:0b   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/12       Up      Down    Down    80:a2:35:ee:a8:0c   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/13       Up      Down    Down    80:a2:35:ee:a8:0d   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/14       Up      Down    Down    80:a2:35:ee:a8:0e   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/15       Up      Down    Down    80:a2:35:ee:a8:0f   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/16       Up      Down    Down    80:a2:35:ee:a8:10   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/17       Up      Down    Down    80:a2:35:ee:a8:11   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/18       Up      Down    Down    80:a2:35:ee:a8:12   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/19       Up      Down    Down    80:a2:35:ee:a8:13   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/20       Up      Down    Down    80:a2:35:ee:a8:14   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/21       Up      Down    Down    80:a2:35:ee:a8:15   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/22       Up      Down    Down    80:a2:35:ee:a8:16   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/23       Up      Down    Down    80:a2:35:ee:a8:17   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/24       Up      Down    Down    80:a2:35:ee:a8:18   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/25       Up      Down    Down    80:a2:35:ee:a8:19   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/26       Up      Down    Down    80:a2:35:ee:a8:1a   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/27       Up      Up      Up      80:a2:35:ee:a8:1b   10G     Full    Fri Nov 20 00:59:11 GMT +0000
2020
ifp-0/0/28       Up      Down    Down    80:a2:35:ee:a8:1c   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/29       Up      Down    Down    80:a2:35:ee:a8:1d   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/30       Up      Down    Down    80:a2:35:ee:a8:1e   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/31       Up      Down    Down    80:a2:35:ee:a8:1f   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/32       Up      Down    Down    80:a2:35:ee:a8:20   10G     Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/33       Up      Down    Down    80:a2:35:ee:a8:21   10G     Full    Mon Nov 16 11:24:09 GMT +0000
```

```
2020
ifp-0/0/34       Up      Down    Down    80:a2:35:ee:a8:22    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/35       Up      Down    Down    80:a2:35:ee:a8:23    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/36       Up      Down    Down    80:a2:35:ee:a8:24    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/37       Up      Down    Down    80:a2:35:ee:a8:25    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/38       Up      Down    Down    80:a2:35:ee:a8:26    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/39       Up      Down    Down    80:a2:35:ee:a8:27    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/40       Up      Down    Down    80:a2:35:ee:a8:28    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/41       Up      Down    Down    80:a2:35:ee:a8:29    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/42       Up      Down    Down    80:a2:35:ee:a8:2a    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/43       Up      Down    Down    80:a2:35:ee:a8:2b    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/44       Up      Down    Down    80:a2:35:ee:a8:2c    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/45       Up      Down    Down    80:a2:35:ee:a8:2d    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/46       Up      Down    Down    80:a2:35:ee:a8:2e    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/47       Up      Down    Down    80:a2:35:ee:a8:2f    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/48       Up      Down    Down    80:a2:35:ee:a8:30    10G    Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/49       Up      Down    Down    80:a2:35:ee:a8:31    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/50       Up      Down    Down    80:a2:35:ee:a8:35    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/51       Up      Down    Down    80:a2:35:ee:a8:39    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
ifp-0/0/52       Up      Up      Up      80:a2:35:ee:a8:3d    100G   Full    Tue Nov 17 14:10:46 GMT +0000
2020
ifp-0/0/53       Up      Up      Up      80:a2:35:ee:a8:41    100G   Full    Fri Nov 20 00:59:12 GMT +0000
2020
ifp-0/0/54       Up      Down    Down    80:a2:35:ee:a8:45    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
cpu-0/0/200      Up      Up      Up      80:a2:35:ee:a8:c8    100G   Full    Mon Nov 16 11:24:11 GMT +0000
2020
cpu-0/0/201      Up      Down    Down    80:a2:35:ee:a8:c9    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
cpu-0/0/202      Up      Down    Down    80:a2:35:ee:a8:ca    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
cpu-0/0/203      Up      Down    Down    80:a2:35:ee:a8:cb    100G   Full    Mon Nov 16 11:24:09 GMT +0000
2020
recycle-0/0/75   Up      Up      Up      80:a2:35:ee:a8:4b    100G   Full    Mon Nov 16 11:24:11 GMT +0000
2020
recycle-0/0/76   Up      Up      Up      80:a2:35:ee:a8:4c    100G   Full    Mon Nov 16 11:24:11 GMT +0000
2020
```

Example 4: List of All Logical Interfaces for All Instances

```
supervisor@rtbrick>LEAF01: op> show interface logical
Interface              Instance            Admin   Link    Oper    Outer VLAN    Inner VLAN   IPv4 Status,MTU
IPv6 Status,MTU
ifl-0/0/10/100         default             Up      Up      Up      -             -            Up,1500
Up,1500
ifl-0/0/10/200         default             Up      Up      Up      200           -            Up,1500
Up,1500
ifl-0/0/10/300         default             Up      Up      Up      300           -            Up,1500
Up,1500
```

Example 5: List of Logical Interfaces for an Instance

```
supervisor@rtbrick: op> show interface logical instance default
Interface            Instance           Admin   Link    Oper    Outer VLAN   Inner VLAN  IPv4 Status,MTU
IPv6 Status,MTU
ifl-0/0/10/100       default            Up      Up      Up      -            -           Up,1500
Up,1500
ifl-0/0/10/200       default            Up      Up      Up      200          -           Up,1500
Up,1500
ifl-0/0/10/300       default            Up      Up      Up      300          -           Up,1500
Up,1500
```

Example 6: List of All Interface Addresses

```
supervisor@rtbrick: op> show interface address
Interface               Instance         IPv4 Address         IPv4 Primary   IPv6 Address
  ifl-0/0/10/100        default        198.51.100.22/24       True          2001:db8:0:100::/32
  ifl-0/0/10/200        default        198.51.100.32/24        True         2001:db8:0:10::/32
  ifl-0/0/10/300        default              -                              2001:db8:0:160::/32
```

To access the Operational State API that corresponds to this CLI, click here.

Interface Details Commands

Syntax:

show interface <option> **detail**

Option	Description
detail	Without any additional option, displays detailed information for all interfaces.
<interface-name> detail	Displays detailed information for an interface.

Example 7: Detailed Information for a Physical Interface and Its Logical Interfaces

```
supervisor@rtbrick: op> show interface ifp-0/0/10 detail
Interface:ifp-0/0/10
  Admin/Link/Operational status: Up/Up/Up
  MRU: 10000
  MTU: 1514
  Interface type: ifp
  Interface index: 8198
  MAC: 7a:4a:14:c0:00:04
  Uptime:  Thu Apr 11 05:57:42 GMT +0000 2024
  Flap count:  2
  Description: Physical interface #4 from node 0, chip 0
  Interface: ifp-0/0/10, Instance: default
    Admin/Link/Operational status: Up/Up/Up
    IPv4/IPv6/MPLS Status: Up/Up/Up
    IPv4/IPv6/MPLS MTU: 1500/1500/1500
    Interface type: Logical Sub interface
    Interface index: 532486
    MAC: 7a:4a:14:c0:00:04
    IPv4 Address             IPv6 Address
    192.0.2.0/24             2001:DB8::/64
```

The following output shows the interface holddown details such as the damping state, transitions, and delay.

```
supervisor@rtbrick.net: cfg> show interface ifp-0/1/20 detail
Interface:ifp-0/1/20
  Admin/Link/Operational status: Up/Up/Up
  Damping Status: stable
  Speed configured: 10G
  Speed maximum: 25G
  Duplex: Full
  Encapsulation mode: ieee
  MRU: 10000
  MTU: 9000
  Maximum frame size: 10000
  Interface type: ethernet
  Interface index: 43265
  MAC: e8:c5:7a:b2:04:b4
  Uptime:  Fri Jul 05 06:19:12 GMT +0000 2024, Holddown up/down delay: 10000ms/0ms
  Flap count:  1
  Holddown transitions:  1
  Description: Physical interface #20 from node 0, chip 1
  Packet statistics:
    Rx packets: 59         Tx packets: 59
    Rx bytes: 7493         Tx bytes: 7493
supervisor@rtbrick>ufi15.q2c.u15.r5.nbg.rtbrick
```

MTU Profile Show Command

Syntax:

show mtu profile <option>

Option	Description
-	Without any additional option, displays detailed information for all MTU profiles.
profile-name <mtu-profile-name>	MTU Profile Name

Example 8: Detailed Information About the MTU Profiles

```
supervisor@rtbrick>LEAF01: op> show mtu profile
```

```
Profile Name                 Type         Size   Action
__default_pppoe__            pppoe        1492   drop
l3IpMtu                      ipv4         1300   drop
l3Ipv6Mtu                    ipv6         1300   drop
portMtu                      physical     1300   drop
portM2                       physical     1400   drop
portM5                       physical     1430   drop
supervisor@rtbrick>LEAF01: op>
```

Example 9: Display Information About the Specified MTU Profile

```
supervisor@rtbrick>LEAF01: op> show mtu profile profile-name l3IpMtu
Profile Name                 Type         Size   Action
l3IpMtu                      ipv4         1300   drop
supervisor@rtbrick>LEAF01: op>
```

Interface Statistics Commands

Syntax:

show interface <option> **statistics**

Option	Description
statistics	Without any additional option, displays statistics information for all interfaces.
<interface-name> statistics	Displays statistics information for an interface.

Example 10: Statistics Information for a Physical Interface and Its Logical Interfaces

```
supervisor@rtbrick>LEAF01: op> show interface ifp-0/0/10 statistics
Interface:  ifp-0/0/10
   Counter              Direction   Unit       Rx         Rx Diff    Rx Rate    Tx         Tx Diff   Tx Rate
   IPv4                 -           Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   IPv6                 -           Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   MPLS                 -           Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   Punt                 -           Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   Miss                 RX          Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   Drops                -           Packets    4995       -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   Error                RX          Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   Error                TX          Packets    47         -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   No Buff              RX          Packets    -          -          -          -          -          -
                                    Bytes      -          -          -          -          -          -
   Traffic Statistics   -           Packets    4995       -          -          68492      -          -
                                    Bytes      489510     -          -          5869876    -          -
   Unicast Statistics   -           Packets    -          -          -          -          -          -
```

```
                                  Bytes      -          -          -          -          -          -
    Broadcast Statistics  -           Packets    -          -          -          -          -          -
                                  Bytes      -          -          -          -          -          -
    Multicast Statistics  -           Packets    -          -          -          -          -          -
                                  Bytes      -          -          -          -          -          -
    Bcm Statistics:
    inOctets:                                511632
    inUcastPkts:                             0
    inNonUcastPkts:                          5016
    inErrors:                                0
    inUnknownProtos:                         0
    outOctets:                               6236484
    outUcastPkts:                            0
    outNonUcastPkts:                         68492
    outErrors:                               0
    etherStatsDropEvents:                    0
    etherStatsMulticastPkts:                 67718
    etherStatsBroadcastPkts:                 5790
    etherStatsUndersizePkts:                 0
    etherStatsFragments:                     0
    etherStatsOversizePkts:                  0
    etherStatsOctets:                        6748116
    etherStatsPkts:                          73508
    etherStatsCollisions:                    0
    etherStatsTXNoErrors:                    68492
    etherStatsRXNoErrors:                    5016
    ifInMulticastPkts:                       5016
    ifOutBroadcastPkts:                      5790
    ifOutMulticastPkts:                      62702
    ifOutBroadcastPkts:                      5790
    bcmReceivedUndersizePkts:                0
    bcmTransmittedUndersizePkts:             5790
    bcmQmxDot1dBasePortDelayExceededDiscards: 0
    bcmQmxDot1dBasePortMtuExceededDiscards:   0
    bcmQmxDot1dTpPortInFrames:               5016
    bcmQmxDot1dTpPortOutFrames:              68492
    bcmQmxEtherStatsPkts64Octets:            5790
    bcmQmxEtherStatsPkts128to255Octets:      24
    bcmQmxEtherStatsPkts256to511Octets:      0
    bcmQmxEtherStatsPkts512to1023Octets:     0
    bcmQmxEtherStatsPkts1024to1518Octets:    0
    bcmQmxEtherRxOversizePkts:               0
    bcmQmxEtherTxOversizePkts:               0
    bcmQmxEtherStatsJabbers:                 0
    bcmQmxEtherStatsCRCAlignErrors:          0
    bcmQmxDot3StatsFCSErrors:                0
    bcmQmxDot3StatsSingleCollisionFrames:    0
    bcmQmxDot3StatsMultipleCollisionFrames:  0
    bcmQmxDot3StatsSQETTestErrors:           0
    bcmQmxDot3StatsDeferredTransmissions:    0
    bcmQmxDot3StatsLateCollisions:           0
    bcmQmxDot3StatsExcessiveCollisions:      0
    bcmQmxDot3StatsInternalMacTransmitErrors: 0
    bcmQmxDot3StatsCarrierSenseErrors:       0
    bcmQmxDot3StatsFrameTooLongs:            0
    bcmQmxDot3StatsInternalMacReceiveErrors: 0
    bcmQmxDot3StatsSymbolErrors:             0
    bcmQmxDot3ControlInUnknownOpcodes:       0
    bcmQmxDot3InPauseFrames:                 0
    bcmQmxDot3OutPauseFrames:                0
    bcmQmxIfHCInOctets:                      511632
    bcmQmxIfHCInUcastPkts:                   0
    bcmQmxIfHCInMulticastPkts:               5016
    bcmQmxIfHCInBroadcastPkts:               0
    bcmQmxIfHCOutOctets:                     6236484
    bcmQmxIfHCOutUcastPkts:                  0
    bcmQmxIfHCOutMulticastPkts:              62702
    bcmQmxIfHCOutBroadcastPckts:             5790
    bcmQmxIeee8021PfcRequests:               0
    bcmQmxIeee8021PfcIndications:            0
    bcmQmxBcmEtherStatsPkts1519to1522Octets: 0
    bcmQmxBcmEtherStatsPkts1522to2047Octets: 0
    bcmQmxBcmReceivedPkts64Octets:           0
    bcmQmxBcmReceivedPkts65to127Octets:      5016
    bcmQmxBcmReceivedPkts128to255Octets:     0
    bcmQmxBcmReceivedPkts256to511Octets:     0
```

```
     bcmQmxBcmReceivedPkts512to1023Octets:     0
     bcmQmxBcmReceivedPkts1024to1518Octets:    0
     bcmQmxBcmReceivedPkts1519to2047Octets:    0
     bcmQmxBcmTransmittedPkts64Octets:         5790
     bcmQmxBcmTransmittedPkts65to127Octets:    62678
     bcmQmxBcmTransmittedPkts128to255Octets:   24
     bcmQmxBcmTransmittedPkts256to511Octets:   0
     bcmQmxBcmTransmittedPkts512to1023Octets:  0
     bcmQmxBcmTransmittedPkts1024to1518Octets: 0
     bcmQmxBcmTransmittedPkts1519to2047Octets: 0
     bcmQmxBcmTransmittedPkts2048to4095Octets: 0
     bcmQmxBcmTransmittedPkts4095to9216Octets: 0

  Logical Interface:  ifl-0/0/10/100, Physical Interface:  ifp-0/0/10
     Counter              Direction   Unit       Rx         Rx Diff    Rx Rate    Tx         Tx Diff   Tx Rate
     IPv4                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     IPv6                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     MPLS                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Punt                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Miss                 RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Drops                -           Packets    4995       -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Error                RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Error                TX          Packets    47         -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     No Buff              RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Traffic Statistics   -           Packets    4995       -          -          68492      -          -
                                      Bytes      489510     -          -          5869876    -          -
     Unicast Statistics   -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Broadcast Statistics -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Multicast Statistics -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Packet Statistics:
       Ingress Forwarded Packets: 1810
       Ingress Forwarded Bytes:   184620
       Ingress Drop Packets:      1
       Ingress Drop Bytes:        102
       Egress Forwarded Packets:  0
       Egress Forwarded Bytes:    0
       Egress Drop Packets:       0
       Egress Drop Bytes:         0
  Logical Interface:  ifl-0/0/10/200, Physical Interface:  ifp-0/0/10
     Counter              Direction   Unit       Rx         Rx Diff    Rx Rate    Tx         Tx Diff   Tx Rate
     IPv4                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     IPv6                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     MPLS                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Punt                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Miss                 RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Drops                -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Error                RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Error                TX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     No Buff              RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Traffic Statistics   -           Packets    -          -          -          6811       -          -
                                      Bytes      -          -          -          573170     -          -
     Unicast Statistics   -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
     Broadcast Statistics -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
```

```
    Multicast Statistics  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Packet Statistics:
      Ingress Forwarded Packets: 0
      Ingress Forwarded Bytes:   0
      Ingress Drop Packets:      0
      Ingress Drop Bytes:        0
      Egress Forwarded Packets:  0
      Egress Forwarded Bytes:    0
      Egress Drop Packets:       0
      Egress Drop Bytes:         0
 Logical Interface:  ifl-0/0/10/300, Physical Interface:  ifp-0/0/10
    Counter               Direction   Unit       Rx         Rx Diff    Rx Rate    Tx         Tx Diff   Tx Rate
    IPv4                  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    IPv6                  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    MPLS                  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Punt                  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Miss                  RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Drops                 -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Error                 RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Error                 TX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    No Buff               RX          Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Traffic Statistics    -           Packets    -          -          -          5902       -          -
                                      Bytes      -          -          -          531180     -          -
    Unicast Statistics    -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Broadcast Statistics  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Multicast Statistics  -           Packets    -          -          -          -          -          -
                                      Bytes      -          -          -          -          -          -
    Packet Statistics:
      Ingress Forwarded Packets: 0
      Ingress Forwarded Bytes:   0
      Ingress Drop Packets:      0
      Ingress Drop Bytes:        0
      Egress Forwarded Packets:  0
      Egress Forwarded Bytes:    0
      Egress Drop Packets:       0
      Egress Drop Bytes:         0
```

Show Port Map Command

The show port map command displays information about a physical interface's mapping with ONL port IDs and optic port IDs.

Syntax:

show port map

Example: Port mapping

```
supervisor@rtbrick>LEAF01: op> show port map
   IFP-Name            Interface-Type       Port-ID   Onlp-Port-ID   Optic-Port-ID
   cpu-0/0/0           cpu                  0         -              -
   ifp-0/0/0           ethernet             1         32             32
```

```
    ifp-0/0/1           ethernet            2         33             33
    ifp-0/0/2           ethernet            3         34             34
    ifp-0/0/3           ethernet            4         35             35
    ifp-0/1/1           ethernet            5         1              1
    ifp-0/1/2           ethernet            9         2              2
    ifp-0/1/3           ethernet            13        3              3
    ifp-0/1/4           ethernet            17        4              4
    ifp-0/1/5           ethernet            21        5              5
    ifp-0/1/6           ethernet            25        6              6
    ifp-0/1/7           ethernet            29        7              7
    ifp-0/1/8           ethernet            33        8              8
    ifp-0/1/9           ethernet            37        9              9
    ifp-0/1/10          ethernet            41        10             10
    ifp-0/1/11          ethernet            45        11             11
    ifp-0/1/12          ethernet            49        12             12
   <...>
```

Show Interface History Command

The show interface history command displays information about the operational status of an interface.

Syntax:

show interface history

Example: Interface history

```
supervisor@rtbrick.net: cfg> show interface history
Interface            Update Time                      Admin Status     Link Status
Oper Status   Speed
ifp-0/0/3         Tue Apr 15 04:32:49 GMT +0000 2025     Up             Up
Up              -
ifp-0/0/2         Tue Apr 15 04:32:49 GMT +0000 2025     Up             Up
Up              -
ifp-0/0/1         Tue Apr 15 04:32:49 GMT +0000 2025     Up             Up
Up              -
ifp-0/0/0         Tue Apr 15 04:32:49 GMT +0000 2025     Up             Up
Up              -
```

Example: Interface history detail

```
supervisor@rtbrick.net: cfg> show interface history detail
Interface: ifp-0/0/3
  Update Time: Tue Apr 15 04:32:49 GMT +0000 2025
    Admin/Link/Operational status: Up/Up/Up
    Hardware Link Status: Down
    Carrier Transitions: 2
    Holddown: enable
Interface: ifp-0/0/2
  Update Time: Tue Apr 15 04:32:49 GMT +0000 2025
    Admin/Link/Operational status: Up/Up/Up
```

```
      Hardware Link Status: Down
      Carrier Transitions: 2
      Holddown: enable
  Interface: ifp-0/0/1
    Update Time: Tue Apr 15 04:32:49 GMT +0000 2025
      Admin/Link/Operational status: Up/Up/Up
      Hardware Link Status: Down
      Carrier Transitions: 2
      Holddown: enable
  Interface: ifp-0/0/0
    Update Time: Tue Apr 15 04:32:49 GMT +0000 2025
      Admin/Link/Operational status: Up/Up/Up
      Hardware Link Status: Down
      Carrier Transitions: 2
      Holddown: enable
```

Interface Clear Commands

Clear commands allow to reset operational states.

Interface Statistics

This command clears interface counters.

Syntax:

clear interface statistics <option>

Option	Description
-	Without any additional option, the command clears the counters for all interfaces.
<interface-name>	Clears the counters for the given interface.

To access the Operational State API that corresponds to this CLI, click here.

Clear Neighbor Interface

This command clears neighbor interface.

Syntax:

clear neighbor <interface-name>

Option	Description
<interface-name>	Clears the given interface.

Example:

```
supervisor@rtbrick.net: cfg> show neighbor
Instance                MAC Address          Interface          IP Address
Dynamic  Entry Time
default                 7a:0e:db:c0:00:00    ifp-0/0/0/10       11.1.1.2
true     Fri May 10 11:00:53
default                 7a:0e:db:c0:00:00    ifp-0/0/0/10
fe80::780e:db00:ac0:0          true     Fri May 10 11:00:09
ip2                     7a:0e:db:c0:00:00    ifp-0/0/0/1        10.1.1.2
true     Fri May 10 11:02:09
ip2                     7a:0e:db:c0:00:00    ifp-0/0/0/1        20.1.1.2
true     Fri May 10 11:00:53
ip2                     11:22:33:44:55:66    ifp-0/0/0/1        25.1.1.1
false    Fri May 10 11:00:02
supervisor@rtbrick.net: cfg> clear neighbor interface ifp-0/0/0/1 ip-address
10.1.1.2
Success: cleared neighbors
supervisor@rtbrick.net: cfg> show neighbor
Instance                MAC Address          Interface             IP Address
Dynamic  Entry Time
default                 7a:0e:db:c0:00:00    ifp-0/0/0/10       11.1.1.2
true     Fri May 10 11:00:53
default                 7a:0e:db:c0:00:00    ifp-0/0/0/10
fe80::780e:db00:ac0:0             true     Fri May 10 11:00:09
ip2                     7a:0e:db:c0:00:00    ifp-0/0/0/1        20.1.1.2
true     Fri May 10 11:00:53
ip2                     11:22:33:44:55:66    ifp-0/0/0/1        25.1.1.1
false    Fri May 10 11:00:02
```

6.2. ARP/ND

6.2.1. ARP/ND Overview

RBFS allows you to set the timer information for ARP and ND routes. The timer information specifies how frequently a device sends messages to its neighbor. RBFS provides a logical interface with which you can configure timers for neighbor routers. These timers are essential for protocols such as ARP and ND.

RBFS supports timers for the following attributes:

Gratuitous ARP Interval

The Gratuitous ARP is sent as a broadcast by a node to communicate its IP address to MAC address mapping on the network. The GARP timer enables you to specify the interval time based on which GARP can be communicated.

Neighbor Probe Interval

RBFS allows you to configure the neighbor probe interval for the specified interface. This attribute is used to ensure that the neighbor is available or not.

Router Advertisement Interval

RBFS allows you to configure router advertisement interval. Router advertisement includes route information to show the network hosts that the router is operational. The router sends these messages periodically within a time range specified with minimum and maximum values. The Router Advertisement Interval timer applies only to IPv6.

Neighbor Scan Interval

It specifies the time interval for neighbor router scanning. It scans the ARP table of a neighbor router to determine which IP addresses are active.

ARP Throttle Interval

ARP throttling is a method of rate limiting of ARP packets and it safeguards the router by limiting too many ARP requests triggered by incoming traffic. RBFS allows you to configure the time interval for ARP throttling.

```
**Support for Permanent ARP Entry on IFLs**
```

RBFS supports the Permanent ARP Entry on IFLs functionality. This functionality allows devices to continually attempt address resolution for IPv4 or Neighbor Discovery for IPv6 on critical IP addresses within a LAN. The device can dynamically resolve the MAC addresses associated with these specified IPs, even if the GARP is missed from the neighbor.

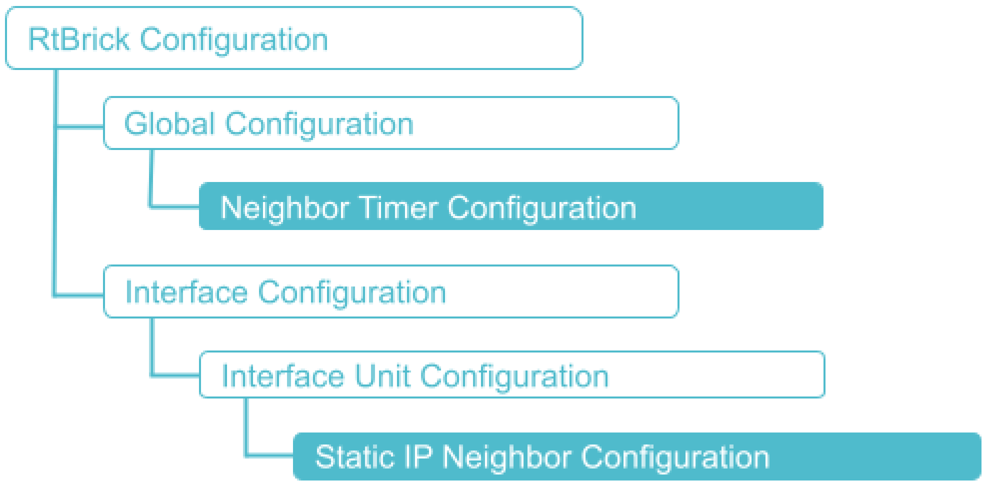
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

6.2.2. ARP/ND Configuration

Configuration Hierarchy

The diagram illustrates the Neighbor Timer configuration hierarchy.


RtBrick Configuration
Global Configuration
Neighbor Timer Configuration
Interface Configuration
Interface Unit Configuration
Static IP Neighbor Configuration

Configuration Syntax and Commands

Neighbor Timer Configuration

The following sections describe the interface configuration syntax and commands.

Syntax:

set global neighbor <attribute> <value>

Attribute	Description
garp-interval	Gratuitous ARP interval. The value is in seconds. Range: 1 to 1000 seconds Default: 60
probe-interval	Neighbor probe interval. The value is in seconds. Range: 1 to 1000 seconds Default: 10
ra-interval	Router advertisement interval. The value is in seconds. Range: 1 to 1000 seconds Default: 10

Attribute	Description
scan interval	Neighbor scan interval. The value is in seconds. Range: 1 to 1000 seconds Default: 60
throttle-interval	ARP throttle interval. The value is in seconds. Range: 1 to 1000 seconds Default: 10

Example: Neighbor timer Configuration

```
{
    "rtbrick-config:neighbor": {
      "garp-interval": 10,
      "probe-interval": 120,
      "scan-interval": 120,
      "throttle-interval": 120,
      "ra-interval": 120
    }
  }
```

Static IP Neighbor Configuration

This section describes configuration options at static IP neighbors.

Syntax:

set interface <interface-name> **unit** <unit-id> **neighbor** <attribute> <value>

Attribute	Description
<interface-name>	Name of the interface. Examples: ifp-0/0/1.
<unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface.
IPv4/IPv6 <ip-address>	Neighbor IPv4 or IPv6 address.
MAC <mac-address>	Neighbor MAC address.

Example: Static IP Neighbor Configuration

```
supervisor@rtbrick>LEAF01: cfg> show config
{
  "data": {
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/1/5",
        "unit": [
          {
            "unit-id": 1,
            "neighbor": {
              "ipv4": [
                {
                  "address4": "198.51.100.10",
                  "mac": "11:11:11:11:11:11"
                }
              ]
            }
          }
        ]
      }
    ]
  }
}
```

Enabling/Disabling GARP on Interface

You can enable or disable the sending of Gratuitous ARP (GARP) on an interface. This section describes enabling or disabling GARP on an interface.

Syntax:

set interface <interface-name> **unit** <unit-id> **gratitous-arp** <enable|disable>

Attribute	Description
interface <interface-name>	Name of the interface. Examples: ifp-0/0/1.
gratitous-arp	Specify enable or disable.

Example: Enabling Gratitous ARP

```
supervisor@rtbrick.net: cfg> set interface hostif-0/0/0 unit 100 gratitous-arp
enable|disable
```

Enabling Permanent ARP Entry on IFLs

You can enable the Permanent ARP Entry on IFLs functionality.

Syntax:

set interface <interface> **unit** <unit id> **neighbor** <*ipv4|ipv6*> <ip address> **dynamic true**

Attribute	Description
<interface-name>	Name of the interface. Example: ifp-0/0/1.
<unit-id>	Create a logical interface (also referred to as a sub-interface) under the physical interface.
neighbor IPv4/IPv6 <ip-address>	Neighbor IPv4 or IPv6 address.
dynamic	Dynamic resolution mode.

Example: Enable Permanent ARP Entry on IFLs

```
{
  "ietf-restconf:data": {
    "rtbrick-config:interface": [
      {
        "name": "ifp-0/0/0",
        "host-if": "S1-1-SN",
        "unit": [
          {
            "unit-id": 10,
            "neighbor": {
              "ipv4": [
                {
                  "address4": "10.2.2.2",
                  "dynamic": "true"
                }
              ]
            }
          }
        ]
      },
      {
        "name": "ifp-0/0/1",
        "host-if": "S1-2-SN"
      },
      {
        "name": "ifp-0/0/2",
        "host-if": "S1-3-L1"
      },
      {
        "name": "ifp-0/0/3",
        "host-if": "S1-4-L1"
      }
    ]
  }
}
```

6.2.3. ARP/ND Operational Commands

Show Commands

Neighbor Timer Show Commands

Syntax:

show neighbor <option>

Option	Description
-	Without any option, the commands display the information for all neighbors.
instance <instance_name>	Displays summary of the specified neighbor instance
IPv4	Displays neighbor summary for the specified IPv4 address.
IPv6	Displays neighbor summary for the specified IPv6 address.
request <…>	Displays information about GARP requests made.

Example 1: Summary of neighbor

```
supervisor@rtbrick>LEAF01: op> show neighbor
Instance                MAC Address          Interface            IP Address                Dynamic  Entry
Time
default                 e4:ed:7a:8e:d5:9d    ifl-0/0/5/1          2001:db8:0:30::
true     Thu Feb 24 02:23:19
```

Example 2: Summary of neighbor instance

```
supervisor@rtbrick>LEAF01: op> show neighbor instance default
Instance                MAC Address          Interface            IP Address
Dynamic  Entry Time
default                 7a:01:bf:60:03:02    ifp-0/2/3/10        2001:db8:0:111::
true     Thu Feb 24 04:57:22
default                 7a:01:bf:60:03:02    ifp-0/2/3/20        2001:db8:0:19::
true     Thu Feb 24 04:57:22
```

Example 3: Summary of the neighbor for the specified address family

```
supervisor@rtbrick>LEAF01: op> show neighbor ipv4
  <cr>
  instance              Instance name
```

```
supervisor@rtbrick>LEAF01: op> show neighbor ipv6
```

```
Instance                MAC Address         Interface            IP Address               Dynamic  Entry
Time
default                 7a:01:bf:60:03:02    ifp-0/2/3/10        2001:db8:0:111::    true     Thu Feb 24
04:57:22
default                 7a:01:bf:60:03:02    ifp-0/2/3/20        2001:db8:0:19::     true     Thu Feb 24
04:57:22
```

Example 4: Summary of the static IP neighbor

```
supervisor@rtbrick>LEAF01: op> show neighbor
Instance                MAC Address          Interface             IP Address
Dynamic  Entry Time
default                 11:11:11:11:11:11    ifl-0/1/5/1           198.51.100.10
true     Fri Feb 25 10:13:04
```

Neighbor Address Resolution

Syntax:

show address resolution <request | response>

Option	Description
Request	Displays the summary of the address resolution request
Response	Displays the summary of address resolution response

Example 1: Summary of the neighbor address resolution request

```
supervisor@rtbrick>LEAF01: op> show address-resolution request
TableName: global.static.1.address.resolution.request
Next Hop                    AFI         SAFI        Instance
2001:db8:0:30::            ipv6        labeled-un  default
```

Example 2: Summary of the neighbor address resolution response

```
supervisor@rtbrick>LEAF01: op> show address-resolution response
TableName: global.static.1.address.resolution.response
IP Address                  Covering Prefix           MAC Address         Interface
2001:db8:0:30::              2001:db8:0:30::/32       e4:ed:7a:8e:d5:9d   ifl-0/0/5/1
```

Neighbor Request

This command is used to display detailed information about various ARP neighbor requests.

Syntax:

show neighbor request <option>

Option	Description
-	Without any option, the commands display the information about all ARP requests made.
type	This option allows you to specify the type of neighbor request.
arp-proxy	Displays information about ARP proxy requests.
arp-reply	Shows ARP reply messages.
arp-request	Displays ARP request messages.
arp-static	Shows static ARP entries.
garp	Displays Gratuitous ARP (GARP) messages.

Example 1: Summary of the request

```
supervisor@rtbrick: cfg> show neighbor request
Interface            IP Address                           Type              Entry
Time
ifl-0/1/10/0         192.168.25.1                         garp              Thu
Aug 22 09:16:12
ifl-0/1/61/10        63.1.1.1                             garp              Thu
Aug 22 09:16:12
ifl-0/1/61/10        63.1.1.2                             arp-request       Thu
Aug 22 09:18:04
ifl-0/1/61/10        63.1.1.2                             arp-static        Thu
Aug 22 09:16:12
ifl-0/1/10/0         fe80::e8c5:7a00:b2:4aa               garp              Thu
Aug 22 09:16:12
ifl-0/1/61/10        fe80::e8c5:7aff:feb2:4dd             garp              Thu
Aug 22 09:16:12
```

Example 2: Summary of the request type GARP

```
supervisor@rtbrick: cfg> show neighbor request type garp
Interface            IP Address                           Type              Entry Time
ifl-0/1/10/0         192.168.25.1                         garp              Thu Aug 22 09:16:12
ifl-0/1/61/10        63.1.1.1                             garp              Thu Aug 22 09:16:12
ifl-0/1/10/0         fe80::e8c5:7a00:b2:4aa               garp              Thu Aug 22 09:16:12
ifl-0/1/61/10        fe80::e8c5:7aff:feb2:4dd             garp              Thu Aug 22 09:16:12
supervisor@rtbrick: cfg> show neighbor request type arp-static
Interface            IP Address                           Type              Entry Time
ifl-0/1/61/10        63.1.1.2                             arp-static        Thu Aug 22 09:16:12
```

Neighbor History

This command is used to display information for deleted or disconnected neighbors. It allows you to retrieve information for the latest 1,000 deleted neighbor entries.

Syntax:

show neighbor history <option>

Option	Description
-	Without any option, the command displays the history of all neighbors.
interface	Displays history of a specified interface.
ipv4	Displays history of a specified IPv4 neighbor.
ipv6	Displays history of a specified IPv6 neighbor.

Example 1: Neighbor history

```
supervisor@rtbrick: cfg> show neighbor history
Instance    MAC Address       Interface         IP Address
Dynamic     Entry Time         Delete Time
default     7a:26:ea:c0:00:00 ifl-0/0/2/10   172.10.16.2                 true
Tue Feb 11 06:06:02   Tue Feb 11 06:07:40
default     7a:26:ea:c0:00:00 ifl-0/0/2/10   2001:db8:3::2               true
Tue Feb 11 06:06:02   Tue Feb 11 06:07:40
default     7a:26:ea:c0:00:00 ifl-0/0/2/10   fe80::7826:eaff:fec0:0       true
Tue Feb 11 06:05:31   Tue Feb 11 06:07:40
default     7a:26:ea:c0:00:01 ifl-0/0/3/2    190.100.6.1                 true
Tue Feb 11 06:06:02   Tue Feb 11 06:07:46
default     7a:26:ea:c0:00:01 ifl-0/0/3/2    fe80::7826:eaff:fec0:1       true
Tue Feb 11 06:05:29   Tue Feb 11 06:07:46
```

Example 2: History of a specified nieghbor interface

```
supervisor@rtbrick: cfg> show neighbor history interface ifl-0/0/2/10
Instance     MAC Address        Interface           IP Address
Dynamic  Entry Time        Delete Time
default      7a:26:ea:c0:00:00  ifl-0/0/2/10    172.10.16.2             true
Tue Feb 11 06:06:02  Tue Feb 11 06:07:40
default      7a:26:ea:c0:00:00  ifl-0/0/2/10    2001:db8:3::2           true
Tue Feb 11 06:06:02  Tue Feb 11 06:07:40
default      7a:26:ea:c0:00:00  ifl-0/0/2/10    fe80::7826:eaff:fec0:0   true
Tue Feb 11 06:05:31  Tue Feb 11 06:07:40
```

6.3. ACLs

6.3.1. ACL Overview

ACL Use Cases

In RBFS, Access Control Lists (ACL) serve multiple purposes:

- Provide security by traffic filtering. This applies to both host and transit traffic. ACLs for traffic filtering are user-defined by configuration.
- Redirecting control traffic to the CPU. Such protocol ACLs also referred to as trap rules, are automatically created by the respective protocol, and do not need to be configured.
- Classifying traffic for differentiated QoS treatment. This is a special form of ACL referred to as a multi-field (MF) classifier. For more information about MF classifiers, please refer to the HQoS Configuration Guide.

ACL Components and Processing

User-defined ACLs consist of rules and ordinals. In case of multiple matching ACL rules, you can use priorities to define the result of the ACL.

- Rules - A rule is a named ACL entry that typically contains one or multiple match criteria and an action.
- Ordinals - An ordinal is solely a numbered configuration object. A rule can consist of multiple ordinals. Ordinals help to structure the configuration. In RBFS, it makes no difference if you configure one rule with multiple ordinals or multiple rules with one ordinal each. Please note ordinals do not define the order of processing.
- Scope - ACLs generally apply globally. In particular, they are not applied to interfaces. You can, however configure an interface as a match criteria.
- Priorities - ACL entry priorities are used to define the processing of multiple matching ACL rules. In RBFS, by default, all ACL entries have the same priority, and there is no specific order. For example, if one ACL rule shall permit ICMP traffic from a specific prefix, and another rule shall deny any other ICMP traffic, it will by default result in a conflict as an ICMP packet matches both rules. To ensure that the more specific rule matches first, you can set its priority to

higher. When the ACL priority value is set to a lower number, priority is higher.

Prefix Lists

A prefix list is a named list of prefixes. Instead of listing multiple individual prefixes in a match rule of the ACL itself, you can reference a list that contains the prefixes, and thereby apply a common action to all matching prefixes. This helps to maintain lists and reuse them in multiple ACL rules.

Prefix lists can be used in ACL for permitting/denying traffic and in Multifield Classifier (MFC) for classifying traffic. This guide describes how to configure prefix lists and apply them in user-defined ACLs as firewall filters and apply prefix lists in MFC for traffic classification. For more information about applying prefix lists to MF classifiers, please refer to the *HQoS configuration Guide*.

When a prefix list is configured and referenced in an ACL, it is internally first added to an intermediate ACL configuration table. For each prefix, one separate rule is added to the final ACL configuration table. This is different from a prefix match in the ACL rule itself that is directly added to the ACL configuration table. A dedicated range of ordinals (200001-4294967295) is reserved to expand ACL rules when using prefix lists. If configured, the priority will be copied from the prefix list ACL configuration to all the expanded ACL rules.

When using prefix lists, the following restrictions apply:

- You cannot configure the same prefix-list name to match the source prefix list and destination prefix list.
- You cannot configure both the source prefix and source prefix list on the same ACL configuration.
- You cannot configure both the destination prefix and destination prefix list on the same ACL configuration.

Supported Platforms

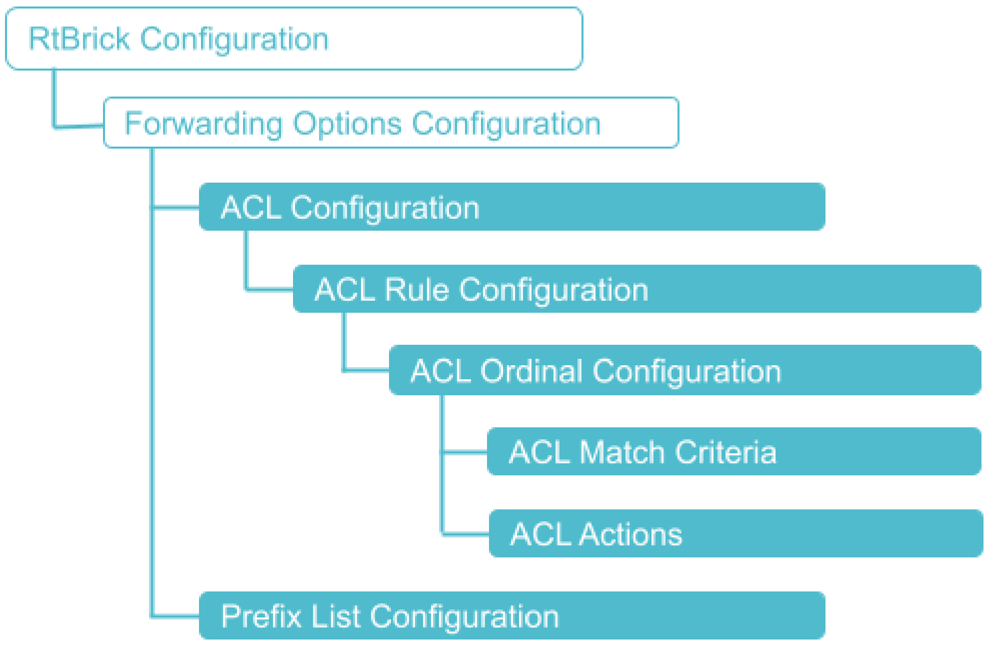
Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

6.3.2. ACL Configuration

For configuring an access control list, you define filter criteria (with match conditions) for the packets and an action for the device to take if the packets match the filtering criteria.

Configuration Hierarchy

The diagram illustrates the ACL configuration hierarchy.



Configuration Syntax and Commands

The following sections describe the ACL configuration syntax and commands.

Configuring ACLs

Syntax:

set forwarding-options acl [**l2** | **l3v4** | **l3v6**] **rule** <name> **ordinal** <number> <option> <attribute> <value>

Options	Description
<name>	Name of the ACL rule.

Options	Description
<number>	Specifies the ordinal number.
match <…>	Match configuration hierarchy. Please refer to section 2.2.1.1 for the ACL match criteria configuration.
action <…>	Action configuration hierarchy. Please refer to section 2.2.1.2 for the ACL actions configuration.
priority <priority>	Specifies the ACL priority value. The default entry priority for user-defined ACLs changes to 500. The configurable ACL entry priority range becomes 100 - 20000. A lower number indicates higher priority.

Configuring ACL Match Criteria

set forwarding-options acl [**l2** | **l3v4** | **l3v6**] **rule** <rulename> **ordinal** <ordinal_value> **match** <attribute> <value>

Attribute	Description
destination-mac <destination-mac>	ACL L2 destination mac match.
destination-ipv4-prefix <destination-ipv4-prefix>	ACL L3 IPv4 destination prefix match.
destination-ipv4-prefix-list <destination-ipv4-prefix-list>	ACL destination IPv4 prefix-list name. You can apply a prefix list that is previously configured. Refer to section Configuring Prefix Lists.
destination-l4-port <destination-l4-port>	ACL L4 destination port match.
destination-ipv4-local true	Indicates whether match support is enabled for all traffic destined for the routers' IP addresses. **Note**: To disable the configuration, use the delete form of the command.
destination-ipv6-prefix <destination-ipv6-prefix>	ACL L3 IPv6 destination prefix match.
destination-ipv6-prefix-list <destination-ipv6-prefix-list>	ACL destination IPv6 prefix-list name. You can apply a prefix list that is previously configured. Refer to section Configuring Prefix Lists.

Attribute	Description
destination-ipv6-local true	Indicates whether match support is enabled for all traffic destined for the routers' IP addresses. To disable the configuration, use the delete form of the command.
direction ingress	ACL L2/L3 direction match. Currently, only the ingress direction is supported.
ethertype <ethertype>	ACL L2 EtherType match.
inner-tag-protocol-id <inner-tag-protocol-id>	ACL L2 inner TPID match.
inner-vlan <inner-vlan>	ACL L2 inner-VLAN match.
inner-vlan-cfi <inner-vlan-cfi>	ACL L2 inner-VLAN CFI match.
inner-vlan-priority <inner-vlan-priority>	ACL L2 inner-VLAN priority match.
interface <interface>	Interface match.
ip-options true	Match if the IPv4 packet has options. Supported value: true.
ip-protocol <protocol>	ACL IP protocol value match such as TCP, UDP, ICMP.
ipv4-dscp <ipv4-dscp>	IPv4 DSCP value.
ipv4-tos <ipv4-tos>	IPv4 ToS value.
ipv6-tc <ipv6-tc>	Codepoint class value.
logical-interface <logical-interface>	Logical interface match.
source-ipv4-prefix <source-ipv4-prefix>	ACL L3 IPv4 source prefix match.
source-ipv4-prefix-list <source-ipv4-prefix-list>	ACL source IPv4 prefix-list name. You can apply a prefix list that is previously configured. Refer to section Configuring Prefix Lists.
source-ipv6-prefix <source-ipv6-prefix>	Configure ACL L3 IPv6 source prefix match.

Attribute	Description
source-ipv6-prefix-list <source-ipv6-prefix-list>	ACL source IPv6 prefix-list name. You can apply a prefix list that is previously configured. Refer to section Configuring Prefix Lists.
source-l4-port <source-l4-port>	ACL L4 source port match.
outer-tag-protocol-id <outer-tag-protocol-id>	ACL L2 outer TPID match.
outer-vlan <outer-vlan>	ACL L2 outer-VLAN match.
outer-vlan-cfi <outer-vlan-cfi>	ACL L2 outer VLAN CFI match.
outer-vlan-priority <outer-vlan-priority>	ACL L2 outer VLAN priority match.
source-mac <source-mac>	ACL L2 source MAC match.
traffic-class <class>	Forward class value. Supported values: class-0 to class-7, class-all.
ttl <ttl>	IPv4 time-to-live value.
match-mpls-traffic true	Match single MPLS label termination. To disable, use the delete form of the command.

Example 1: Layer 2 Match Configuration

```
{
    "rtbrick-config:acl": {
       "l2": {
          "rule": [
            {
              "rule-name": "a10nsp-drop-lag-2",
              "ordinal": [
                {
                  "ordinal-value": 1,
                  "match": {
                    "direction": "ingress",
                    "interface": "lag-2",
                    "outer-vlan-priority": 1
                  },
                  "action": {
                    "drop": "true",
                    "statistics": "true"
                  }
                },
```

Example 2: Layer 3 IPv4 Match Configuration

```
{
    "rtbrick-config:acl": {
      "l3v4": {
        "rule": [
          {
            "rule-name": "rtb_firewall_two",
            "ordinal": [
              {
                "ordinal-value": 1000,
                "match": {
                  "direction": "ingress",
                  "source-ipv4-prefix": "198.51.100.50/24",
                  "source-l4-port": 8080
                },
                "action": {
                  "drop": "true"
                }
              }
            ]
          },
          {
            "rule-name": "rule2",
            "ordinal": [
              {
                "ordinal-value": 5,
                "match": {
                  "direction": "ingress",
                  "interface": "ifp-0/0/1"
                }
              }
            ]
          }
        ]
      }
    }
  }
```

Example 3: Layer 3 IPv6 Match Configuration

```
{
    "rtbrick-config:l3v6": {
      "rule": [
        {
          "rule-name": "rtb_firewall_two",
          "ordinal": [
            {
              "ordinal-value": 1000,
              "match": {
                "direction": "ingress",
                "source-ipv6-prefix": "2001:db8:0:11::/32",
                "source-l4-port": 8080
              },
              "action": {
                "permit": "true"
              }
          }
        ]
      }
    ]
  }
}
```

Example 4: Match support for all traffic destined for any of the router’s IP addresses

```
{
    "rtbrick-config:acl": {
      "l3v4": {
        "rule": [
          {
            "rule-name": "rule4",
            "ordinal": [
              {
                "ordinal-value": 4,
                "match": {
                  "direction": "ingress",
                  "destination-ipv4-local": "true"
                },
                "action": {
                  "drop": "true"
                }
              }
            ]
          }
        ]
      },
      "l3v6": {
        "rule": [
          {
            "rule-name": "rule2",
            "ordinal": [
              {
                "ordinal-value": 2,
                "match": {
                  "direction": "ingress",
                  "destination-ipv6-local": "true"
                },
                "action": {
                  "drop": "true"
                }
              }
            ]
          }
        ]
      }
    }
  }
```

Configuring ACL Actions

Syntax:

set forwarding-options acl [**l3v4** | **l3v6**] **rule** <rulename> **ordinal** <ordinal_value> **action** <attribute> <value>

Attribute	Description
drop true	If the ACL rule specifies drop true, the system will discard any packets that match that rule. Use the delete form of the command for the system to ignore the rule.
permit true	If the ACL rule specifies permit true, the system forwards traffic matching that rule. Use the delete form of the command for the system to ignore the rule.
action statistics true	Configure action, enable statistics. Use the delete form of the command to disable the configuration. A limited number of counter resources are available in a common pool for user-defined ACLs, protocol ACLs, L3, and L2X logical interfaces.
forward-class <class>	Specifies forward class value (class-0 to class-7, class-all)
mirror <mirror>	Specifies ACL action mirror name. Currently, ACLs with mirror actions are not supported.
capture true	You can enable the capture action when using the RBFS built-in capture feature with an ACL to more granularly specify the traffic to be captured. To disable, use the delete form of the command. For more information and an example, refer to the *RBFS NOC Troubleshooting Guide*.
policer-name <policer-name>	Specifies policer profile name.

Attribute	Description
redirect-to-cpu true	Configure action, redirect packets to CPU. To disable, use the delete form of the command.

Example

```
{
            "rule-name": "rtb_firewall_two",
            "ordinal": [
              {
                "ordinal-value": 1000,
                "match": {
                  "direction": "ingress",
                  "source-ipv4-prefix": "198.51.100.50/24",
                  "source-l4-port": 8080
                },
                "action": {
                  "drop": "true"
                }
              }
            ]
          }
```

Configuring Prefix Lists

Configuring IPv4/IPv6 Prefix List for ACL and Multifield Classifier

Syntax:

set forwarding-options prefix-list <prefix-list-name> <attribute> <value>

Attribute	Description
<prefix-list-name>	Name of the prefix list, which will be later used to attach with ACL configuration.
ipv4-prefix <ipv4_prefix>	Specifies the IPv4 prefix address.
ipv6-prefix <ipv6_prefix>	Specifies the IPv6 prefix address.

Example 1: Prefix List Configuration

```
supervisor@rtbrick>LEAF01: op> show config forwarding-options prefix-list
{
    "rtbrick-config:prefix-list": [
      {
        "prefix-list-name": "ipv4-list",
        "ipv4-prefix": [
          {
            "ipv4-prefix": "198.51.100.50/24"
          },
          {
            "ipv4-prefix": "198.51.101.60/24"
          },
          {
            "ipv4-prefix": "198.51.102.70/24"
          }
        ]
      }
    ]
  }
```

Example 2: Using Prefix-list in Multifield-Classifier

```
supervisor@rtbrick>LEAF01: op> show config forwarding-options prefix-list pta-
iptv-multicast
{
  "rtbrick-config:prefix-list": [
    {
      "prefix-list-name": "ipv4-list",
      "ipv4-prefix": [
          {
            "ipv4-prefix": "198.51.100.50/24"
          },
          {
            "ipv4-prefix": "198.51.101.60/24"
          },
          {
            "ipv4-prefix": "198.51.102.70/24"
          }
      ]
    }
  ]
}
```

Example 3: Viewing Multifield-Classifier Details

```
supervisor@rtbrick>LEAF01: op> show config forwarding-options class-of-service
multifield-classifier acl l3v4 rule pta-triple-play-8queues ordinal 6000
{
  "rtbrick-config:ordinal": [
    {
      "ordinal-value": 6000,
      "match": {
        "destination-ipv4-prefix-list": "ipv4-list"
      },
      "action": {
        "forward-class": "class-1",
        "remark-codepoint": 248
      }
    }
  ]
}
```

6.3.3. ACL Operational Commands

ACL Show Commands

Syntax:

show acl <option>

Option	Description
-	Without any option, this command displays brief information about the access control list (ACL).
detail	Displays detailed information about the access-control list (ACL).
type <acl_type>	Displays detailed information for the specified ACL type.
rule <acl-rule-name>	Displays detailed information for the specified ACL rule name.
ordinal <ordinal>	Displays detailed information for the specified Ordinal
type <acl_type> rule <rule_name>	Displays detailed information for the specified ACL Type and Rule name
type <acl_type> ordinal <ordinal>	Displays detailed information for the specified ACL Type and Ordinal
rule <rule_name> type <acl_type>	Displays detailed information for the specified ACL Type and Rule name
rule <rule_name> ordinal <ordinal>	Displays detailed information for the specified Rule name and Ordinal
ordinal <ordinal> type <acl_type>	Displays detailed information for the specified ACL Type and Ordinal
ordinal <ordinal> rule <rule_name>	Displays detailed information for the specified Rule name and Ordinal
type <acl_type> rule <rule_name> ordinal <ordinal>	Displays detailed information for the specified Ordinal, ACL Type and Rule name

Option	Description
type <acl_type> ordinal <ordinal> rule <rule_name>	Displays detailed information for the specified Ordinal, ACL Type and Rule name
rule <rule_name> ordinal <ordinal> type <acl_type>	Displays detailed information for the specified Ordinal, ACL Type and Rule name
rule <rule_name> type <acl_type> ordinal <ordinal>	Displays detailed information for the specified Ordinal, ACL Type and Rule name
ordinal <ordinal> rule <rule_name> type <acl_type>	Displays detailed information for the specified Ordinal, ACL Type and Rule name
ordinal <ordinal> type <acl_type> rule <rule_name>	Displays detailed information for the specified Ordinal, ACL Type and Rule name

Example 1: Show information about ACLs

```
supervisor@rtbrick>LEAF01: op> show acl
ACL                            Ordinal    Type         Attach Point
rule4                          4          l3v4         -
                               8          l3v4         -
lldp.ifp-0/0/0.trap.rule       -          l2           ifp-0/0/0
lldp.ifp-0/1/0.trap.rule       -          l2           ifp-0/1/0
lldp.ifp-0/1/1.trap.rule       -          l2           ifp-0/1/1
lldp.ifp-0/1/4.trap.rule       -          l2           ifp-0/1/4
lldp.ifp-0/1/5.trap.rule       -          l2           ifp-0/1/5
lldp.ifp-0/1/6.trap.rule       -          l2           ifp-0/1/6
lldp.ifp-0/1/12.trap.rule      -          l2           ifp-0/1/12
lldp.ifp-0/1/13.trap.rule      -          l2           ifp-0/1/13
lldp.ifp-0/1/22.trap.rule      -          l2           ifp-0/1/22
lldp.ifp-0/1/23.trap.rule      -          l2           ifp-0/1/23
```

Example 2: Show detailed information about ACLs

```
supervisor@rtbrick>LEAF01: op> show acl detail
Rule: rule4
  ACL type: l3v4
  Ordinal: 4
    Match:
      Direction: ingress
      Source IPv4 prefix: 198.51.100.35/24
    Action:
      Drop: True
    Result:
      Trap ID: User Defined
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    4           0           4
      Bytes      424         0           424
  Ordinal: 8
    Match:
      Direction: ingress
      Source IPv4 prefix: 198.51.100.45/24
    Action:
      Drop: True
    Result:
      Trap ID: User Defined
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    9           0           9
      Bytes      990         0           990
Rule: lldp.ifp-0/0/0.trap.rule
  ACL type: l2
  Ordinal: -
    Match:
      Attachment point: ifp-0/0/0
      Direction: ingress
      Destination MAC: 01:80:c2:00:00:0e
    Action:
      Redirect to CPU: True
    Result:
      Trap ID: LLDP
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    105         105         0
      Bytes      12915       12915       0
Rule: lldp.ifp-0/1/0.trap.rule
  ACL type: l2
  Ordinal: -
    Match:
      Attachment point: ifp-0/1/0
      Direction: ingress
      Destination MAC: 01:80:c2:00:00:0e
    Action:
      Redirect to CPU: True
    Result:
      Trap ID: LLDP
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    220         220         0
      Bytes      19140       19140       0
```

Example 3: Show detailed information for a specified ACL Rule

```
supervisor@rtbrick>LEAF01: op> show acl rule4
Rule: rule4
  ACL type: l3v4
  Ordinal: 4
    Match:
      Direction: ingress
      Source IPv4 prefix: 198.51.100.35/24
    Action:
      Drop: True
    Result:
      Trap ID: User Defined
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    4           0           4
      Bytes      424         0           424
  Ordinal: 8
    Match:
      Direction: ingress
      Source IPv4 prefix: 198.51.100.45/24
    Action:
      Drop: True
    Result:
      Trap ID: User Defined
    Statistics:
      Units      Total       Accepted    Dropped
      Packets    9           0           9
      Bytes      990         0           990
```

ACL Statistics Commands

ACL statistics are currently not supported for PIM, IGMP, and L2TP protocol traffic.

Syntax:

show acl <option> **statistics**

Option	Description
statistics	Displays ACL statistics information
<acl-name> statistics	Displays ACL statistics information for the specified ACL
type <acl_type> statistics	Displays ACL statistics information for the specified ACL type
rule <rule_name> statistics	Displays ACL statistics information for the specified rule name
ordinal <ordinal> statistics	Displays ACL statistics information for the specified ordinal
type <acl_type> rule <rule_name> statistics	Displays ACL statistics information for the specified ACL type and rule name
type <acl_type> ordinal <ordinal> statistics	Displays ACL statistics information for the specified ACL type and ordinal

Option	Description
rule <rule_name> type <acl_type> statistics	Displays ACL statistics information for the specified ACL type and rule name
rule <rule_name> ordinal <ordinal> statistics	Displays ACL statistics information for the specified rule name and ordinal
ordinal <ordinal> type <acl_type> statistics	Displays ACL statistics information for the specified ACL type and ordinal
ordinal <ordinal> rule <rule_name> statistics	Displays ACL statistics information for the specified rule name and ordinal
<acl_type> rule <rule_name> ordinal <ordinal> statistics	Displays ACL statistics information for the specified ordinal, ACL type and rule name
type <acl_type> ordinal <ordinal> rule <rule_name> statistics	Displays ACL statistics information for the specified ordinal, ACL type and rule name
rule <rule_name> ordinal <ordinal> type <acl_type> statistics	Displays ACL statistics information for the specified ordinal, ACL type and rule name
rule <rule_name> type <acl_type> ordinal <ordinal> statistics	Displays ACL statistics information for the specified ordinal, ACL type and rule name
ordinal <ordinal> rule <rule_name> type <acl_type> statistics	Displays ACL statistics information for the specified ordinal, ACL type and rule name
ordinal <ordinal> type <acl_type> rule <rule_name> statistics	Displays ACL statistics information for the specified ordinal, ACL type and rule name

Example 1: Display ACL statistics information

```
supervisor@rtbrick>LEAF01: op> show acl statistics
ACL                                 Units      Total        Accepted     Dropped
rule4                               Packets    4            0            4
                                    Bytes      424          0            424
rule4                               Packets    9            0            9
                                    Bytes      990          0            990
lldp.ifp-0/0/0.trap.rule            Packets    107          107          0
                                    Bytes      13161        13161        0
lldp.ifp-0/1/0.trap.rule            Packets    221          221          0
```

```
                                Bytes      19227       19227       0
lldp.ifp-0/1/1.trap.rule        Packets    221         221         0
                                Bytes      19227       19227       0
lldp.ifp-0/1/4.trap.rule        Packets    214         214         0
                                Bytes      31672       31672       0
lldp.ifp-0/1/5.trap.rule        Packets    214         214         0
                                Bytes      31672       31672       0
lldp.ifp-0/1/6.trap.rule        Packets    214         214         0
                                Bytes      31672       31672       0
lldp.ifp-0/1/12.trap.rule       Packets    107         107         0
                                Bytes      13375       13375       0
lldp.ifp-0/1/13.trap.rule       Packets    107         107         0
                                Bytes      13375       13375       0
lldp.ifp-0/1/22.trap.rule       Packets    107         107         0
                                Bytes      13375       13375       0
lldp.ifp-0/1/23.trap.rule       Packets    107         107         0
                                Bytes      13375       13375       0
```

Example 2: Display ACL statistics information for the specified ACL

```
supervisor@rtbrick>LEAF01: op> show acl rule4 statistics
ACL         Units      Total       Accepted    Dropped
rule4       Packets    4           0           4
            Bytes      424         0           424
rule4       Packets    9           0           9
            Bytes      990         0           990
```

ACL Clear Commands

Clear commands allow resetting operational states.

Clear ACL Statistics

Syntax:

clear acl <options>

Option	**Description**
statistics	Clears all the ACL statistics.
type <acl_type> statistics	Clears all ACL statistics for the specified ACL type
rule <rule_name> statistics	Clears all ACL statistics for the specified rule
ordinal <ordinal> statistics	Clears all ACL statistics for the specified ordinal
type <acl_type> rule <rule_name> statistics	Clears all ACL statistics for the specified ACL type and rule

Option	Description
type <acl_type> ordinal <ordinal> statistics	Clears all ACL statistics for the specified ACL type and ordinal
rule <rule_name> type <acl_type> statistics	Clears all ACL statistics for the specified ACL type and rule
rule <rule_name> ordinal <ordinal> statistics	Clears all ACL statistics for the specified rule and ordinal
ordinal <ordinal> type <acl_type> statistics	Clears all ACL statistics for the specified ACL Type and Ordinal
ordinal <ordinal> rule <rule_name> statistics	Clears all ACL statistics for the specified Rule name and Ordinal
type <acl_type> rule <rule_name> ordinal <ordinal> statistics	Clears all ACL statistics for the specified Ordinal, ACL Type and Rule name
type <acl_type> ordinal <ordinal> rule <rule_name> statistics	Clears all ACL statistics for the specified Ordinal, ACL Type and Rule name
rule <rule_name> ordinal <ordinal> type <acl_type> statistics	Clears all ACL statistics for the specified Ordinal, ACL Type and Rule name
rule <rule_name> type <acl_type> ordinal <ordinal> statistics	Clears all ACL statistics for the specified Ordinal, ACL Type and Rule name
ordinal <ordinal> rule <rule_name> type <acl_type> statistics	Clears all ACL statistics for the specified Ordinal, ACL Type and Rule name
ordinal <ordinal> type <acl_type> rule <rule_name> statistics	Clears all ACL statistics for the specified Ordinal, ACL Type and Rule name

Example: Clearing ACL Statistics of a specified ACL Rule

```
supervisor@rtbrick>LEAF01: op> clear acl rule lldp.ifp-0/0/44.trap.rule statistics
Success : command success
supervisor@rtbrick>LEAF01: op>
```

6.4. Port Mirroring

6.4.1. Port Mirroring Overview

Port Mirroring is a method of monitoring network traffic. When you enable port mirroring, the switch sends a copy of all network packets seen on one port to another port, where the packet can be analyzed.

Inbound Mirroring

Inbound mirroring is defined per In-Port, or per In-Port x VLAN. Configurations for six distinct VLAN tags, for any other VLAN tag, and for packets without VLAN tags are supported. The ingress mirroring can be sampled by specifying a probability that a matching packet will be mirrored.

Outbound Mirroring

Outbound mirroring is defined per Out-Port, or per Out-Port x VLAN tag. Configurations for seven distinct VLAN tags are supported.

Guidelines and Limitations

- Up to 15 mirror profiles can be configured.
- The same mirror resources are used for Lawful Interception (LI) and Port Mirroring.
- You can configure a CPU port as destination physical interface port; but if heavy traffic is mirrored, it may impact system performance.
- If physical interface/logical interface goes down, mirror configuration will not be deleted automatically. You need to delete the mirror configuration explicitly.
- Before creating logical interface mirroring, the source logical interface should exist.
- The logical interface should not be deleted during mirroring.
- If you want to mirror traffic to CPU, enable the control plane security features. For more refer, see the *Control Plane Security Guide*.
- Since this is a debugging tool, the save and reload functionality is not supported.

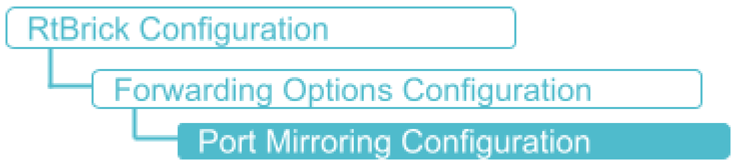
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

6.4.2. Port Mirroring Configuration

Configuration Hierarchy

The diagram illustrates the port mirroring configuration hierarchy.


RtBrick Configuration
Forwarding Options Configuration
Port Mirroring Configuration

Configuration Syntax and Commands

The following sections describe the port mirroring configuration syntax and commands.

Syntax:

set forwarding-options mirror <name> [**source** | **destination**] <attribute> <value>

Attribute	Value
<name>	Name for mirror configuration
acl true	Configure source as ACL. Use delete form of the command to disable the configuration.
direction [egress | ingress]	Configure traffic direction ingress/egress.
interface <interface>	Specifies the physical interface name.
logical-interface <logical-interface>	Configure source logical interface name.

Example 1: Mirroring one physical interface traffic to another physical interface

```
{
    "rtbrick-config:mirror": [
      {
        "name": "MIRROR1",
        "destination": {
          "interface": "ifp-0/0/4"
        },
        "source": {
          "direction": "ingress",
          "interface": "ifp-0/0/2"
        }
      }
    ]
  }
```

Example 2: Mirroring Traffic to CPU

```
{
    "rtbrick-config:mirror": [
      {
        "name": "mirror1",
        "destination": {
          "interface": "cpu-0/0/200"
        },
        "source": {
          "direction": "ingress",
          "interface": "ifp-0/0/52"
        }
      }
    ]
  }
```

6.4.3. Port Mirroring Operational Commands

Capturing Mirror Traffic

After you configure mirroring to CPU by using the commands above, you can use the **capture** command to capture the mirror traffic.

Syntax:

capture mirror file <file_name> [**start** | **stop**]

Attribute	Value
<file_name>	Name of the file where mirror traffic is captured

Example 1: Starting and stopping mirror traffic to a file

```
root@rtbrick: cfg> capture mirror file test.pcap start
root@rtbrick: cfg> capture mirror file test.pcap stop
```

Show Commands

show capture sessions

Syntax:

show capture sessions

Option	Description
-	Without any option, the commands displays the FIB packet Capture sessions.

Example 1: Summary of FIB packet Capture sessions

```
supervisor@rtbrick: op> show capture sessions
Interface                      Direction   File                                       Context
ifp-0/0/1                      BOTH        -                                          45
```

6.5. OAM Support

6.5.1. OAM Overview

Operations, Administration and Management (OAM) are the processes, activities, tools, and standards involved with performing operational, administrative, and management tasks. RBFS provides the following OAM features that enable you troubleshoot RtBrick software:

- IP ping
- IP traceroute
- IP ping on an MPLS transport
- IP traceroute on an MPLS transport

MPLS ping and traceroute will be supported in the later releases

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

6.5.2. OAM Configuration

Guidelines

- Execute the OAM commands in the **operation** mode of the CLI.

```
admin@rtbrick:~$ cli
admin@rtbrick: cfg> switch-mode operation
Activating syntax mode : op [operation]
admin@rtbrick: op>
```

IP Ping

The IP ping utility is used to check the reachability of an IP address.

IP ping in default instance

The **ping** command allows you to ping to a destination to see if a networked device is reachable.

Syntax

```
ping <destination-ip> [source-interface <interface>] count <count> interval
<interval> size <size> source-ip <source-ip> ttl <ttl> tos <tos>
```

Command Parameters

destination-ip	ipv4 destination address
interface	source interface
count	count of the ping packet (default 5)
do-not-fragment	set the do-not-fragment (DF) bit in IP header
interval	interval between the packets (default 1 sec)
size	size of the ping packet (default 60)

destination-ip	**ipv4 destination address**
source-ip	IPv4 source address
ttl	Time-to-live to be used in the packet
tos	Type of Service (TOS) to be used in the packet

Example

```
admin@rtbrick: op> ping 198.51.100.111
68 bytes from 198.51.100.111: icmp_seq=1 ttl=64 time=11.8707 ms
68 bytes from 198.51.100.111: icmp_seq=2 ttl=64 time=1.9824 ms
68 bytes from 198.51.100.111: icmp_seq=3 ttl=64 time=5.0726 ms
68 bytes from 198.51.100.111: icmp_seq=4 ttl=64 time=5.6529 ms
68 bytes from 198.51.100.111: icmp_seq=5 ttl=64 time=10.6588 ms
Statistics: 5 sent, 5 received, 0% packet loss
```

IP ping in specific instance and afi/safi

This command allows you to ping to a destination in a particular VRF.

Syntax

```
ping <destination-ip> [instance <instance-name> afi <afi> safi<safi>] [source-
interface <interface>] count <count> interval <interval> size <size> source-ip
<source-ip> ttl <ttl> tos <tos>
```

Command Parameters

destination-ip	ipv4 destination address
instance-name	instance on which ping has to be executed
afi	IPv4 Address Family Identifier (AFI)
safi	Subsequent address family identifier (SAFI)
interface	source interface
count	count of the ping packet (default 5)
do-not-fragment	set the do-not-fragment (DF) bit in IP header
interval	interval between the packets (default 1 sec)
size	size of the ping packet (default 60)
source-ip	IPv4 source address

ttl	Time-to-live to be used in the packet
tos	Type of Service (TOS) to be used in the packet

The afi/safi attributes are optional; if not specified, afi would be ipv4 and safi would be unicast.

Example

```
admin@rtbrick: op> ping 198.51.100.80 instance ip2vrf afi ipv4 safi labeled-
unicast
68 bytes from 198.51.100.80: icmp_seq=1 ttl=64 time=18.1306 ms
68 bytes from 198.51.100.80: icmp_seq=2 ttl=64 time=32.1058 ms
68 bytes from 198.51.100.80: icmp_seq=3 ttl=64 time=19.8205 ms
68 bytes from 198.51.100.80: icmp_seq=4 ttl=64 time=20.0144 ms
68 bytes from 198.51.100.80: icmp_seq=5 ttl=64 time=32.0085 ms
Statistics: 5 sent, 5 received, 0% packet loss
```

IPv6 Ping

The IPv6 ping utility is used to check the reachability of an IPv6 address.

IPv6 ping in default instance

The **ping** command allows you to ping to an IPv6 destination to see if a networked device is reachable.

Syntax

```
ping <destination-ipv6> [source-interface <interface>] count <count> interval
<interval> size <size> source-ip <source-ipv6> ttl <ttl> tos <tos>
```

Command Parameters

destination-ipv6	ipv6 destination address
interface	source interface
count	count of the ping packet (default 5)
do-not-fragment	set the do-not-fragment (DF) bit in IP header
interval	interval between the packets (default 1 sec)
size	size of the ping packet (default 60)

source-ipv6	IPv6 source address
ttl	Time-to-live to be used in the packet
tos	Type of Service (TOS) to be used in the packet

Example

```
admin@rtbrick: op> ping 2001:db8:0:42::
68 bytes from 2001:db8:0:42::: icmp_seq=1 ttl=64 time=.0503 ms
68 bytes from 2001:db8:0:42::: icmp_seq=2 ttl=64 time=.0321 ms
68 bytes from 2001:db8:0:42::: icmp_seq=3 ttl=64 time=.0314 ms
68 bytes from 2001:db8:0:42::: icmp_seq=4 ttl=64 time=.0325 ms
68 bytes from 2001:db8:0:42::: icmp_seq=5 ttl=64 time=.0354 ms
Statistics: 5 sent, 5 received, 0% packet loss
```

IPv6 ping in specific instance and afi/safi

This command allows you to ping to an IPv6 destination in a particular VRF.

Syntax

```
ping <destination-ipv6> [instance <instance-name> afi<afi> safi<safi>] [source-
interface <interface>] count <count> interval <interval> size <size> source-ip
<source-ipv6> ttl <ttl> tos <tos>
```

Command Parameters

destination-ipv6	ipv6 destination address
instance-name	instance on which ping has to be executed
afi	IPv4 Address Family Identifier (AFI)
safi	Subsequent address family identifier (SAFI)
interface	source interface
count	count of the ping packet (default 5)
do-not-fragment	set the do-not-fragment (DF) bit in IP header
interval	interval between the packets (default 1 sec)
size	size of the ping packet (default 60)
source-ipv6	IPv6 source address
ttl	Time-to-live to be used in the packet

tos	Type of Service (TOS) to be used in the packet

The afi/safi attributes are optional; if not specified, afi would be ipv6 and safi would be unicast.

Example

```
admin@rtbrick: op> ping 2001:db8:0:42:: instance abc afi ipv6 safi labeled-unicast
68 bytes from 2001:db8:0:42::: icmp_seq=1 ttl=64 time=.0503 ms
68 bytes from 2001:db8:0:42::: icmp_seq=2 ttl=64 time=.0321 ms
68 bytes from 2001:db8:0:42::: icmp_seq=3 ttl=64 time=.0314 ms
68 bytes from 2001:db8:0:42::: icmp_seq=4 ttl=64 time=.0325 ms
68 bytes from 2001:db8:0:42::: icmp_seq=5 ttl=64 time=.0354 ms
Statistics: 5 sent, 5 received, 0% packet loss
```

IP traceroute

IP traceroute in default instance

This command allows you to traceroute to a particular IP destination.

Syntax

```
traceroute <destination-ip> [source-interface <interface>] repeat <repeat>
interval <interval> size <pktsize> source-ip <source-ip> maxhop <maxhop>
```

destination-ip	ipv4 destination address
interface	source interface
repeat	no of packets for each hop (default 3)
interval	interval between the packets (default 1 sec)
pktsize	size of the traceroute packet (default 60)
source-ip	source IP address
maxhop	max number of hops before the TTL expires (default 30)

Example

```
admin@rtbrick: op> traceroute 198.51.100.80
traceroute to 198.51.100.80, 30 hops max, 60 byte packets
1    198.51.100.90    39.401 ms       19.919 ms       20.074 ms
```

```
2    198.51.100.80    55.544 ms      36.765 ms      45.989 ms
```

IP traceroute in specific instance and afi/safi

This command allows you to traceroute to a particular IP destination in a specific VRF.

Syntax

```
traceroute <destination-ip> [instance <instance-name> afi <afi> safi<safi>]
[source-interface <interface>] repeat <repeat> interval <interval> size <pktsize>
source-ip <source-ip> maxhop <maxhop>
```

destination-ip	ipv4 destination address
instance-name	instance on which traceroute has to be executed
afi	IPv4 Address Family Identifier (AFI)
safi	Subsequent address family identifier (SAFI)
interface	source interface
repeat	no of packets for each hop (default 3)
interval	interval between the packets (default 1 sec)
pktsize	size of the traceroute packet (default 60)
source-ip	source IP address
maxhop	max number of hops before the TTL expires (default 30)

The afi/safi attributes are optional; if not specified, afi would be ipv4 and safi would be unicast.

Example

```
supervisor@S1-STD-28-2901>bm13-tst.fsn.rtbrick.net: cfg> traceroute 198.51.100.80
instance default afi ipv4 safi labeled-unicast source-interface
 ifl-0/0/0/1 source-ip 198.51.100.55
traceroute to 198.51.100.80 30 hops max, 60 byte packets
1    198.51.100.12    4.961 ms    .421 ms    .503 ms
     MPLS Label=20071  Exp=0  TTL=1  S=1
2    198.51.100.80    .995 ms    6.456 ms    .813 ms
```

For the MPLS transport, it displays the ingress labels in each hop.

IPv6 traceroute

IPv6 traceroute in default instance

This command allows you to traceroute to a particular IP destination.

Syntax

```
traceroute <destination-ipv6> [source-interface <interface>] repeat <repeat>
interval <interval> size <pktsize> source-ip <source-ipv6> maxhop <maxhop>
```

destination-ipv6	ipv6 destination address
interface	source interface
repeat	no of packets for each hop (default 3)
interval	interval between the packets (default 1 sec)
pktsize	size of the traceroute packet (default 60)
source-ip	IPv6 source address
maxhop	max number of hops before the TTL expires (default 30)

Example

```
admin@rtbrick: op> traceroute 2001:db8:0:75::
traceroute to 2001:db8:0:75:: 30 hops max, 60 byte packets
1 2001:db8:0:90:: 21.247 ms 20.232 ms 20.052 ms
2 2001:db8:0:75:: 50.124 ms 59.822 ms 40.032 ms
```

IPv6 traceroute in specific instance and afi/safi

This command allows you to traceroute to a particular IPv6 destination in a specific VRF.

Syntax

```
traceroute <destination-ipv6> [instance <instance-name> afi<afi> safi<safi>]
[source-interface <interface>] repeat <repeat> interval <interval> size <pktsize>
source-ip <source-ipv6> maxhop <maxhop>
```

destination-ipv6	ipv6 destination address
instance-name	instance on which traceroute has to be executed
afi	IPv4 Address Family Identifier (AFI)
safi	Subsequent address family identifier (SAFI)
interface	source interface
repeat	no of packets for each hop (default 3)
interval	interval between the packets (default 1 sec)
pktsize	size of the traceroute packet (default 60)
source-ip	IPv6 source address
maxhop	max number of hops before the TTL expires (default 30)

The afi/safi attributes are optional; if not specified, afi would be ipv6 and safi would be unicast.

Example

```
supervisor@S1-STD-28-2901>bm13-tst.fsn.rtbrick.net: cfg> traceroute
2001:db8:0:75:: instance default afi ipv6 safi labeled-unicast source-interface
 ifl-0/0/0/1 source-ip 2001:db8:0:11::
traceroute to 2001:db8:0:75:: 30 hops max, 60 byte packets
1    2001:db8:0:90::     6.782 ms   .306 ms   6.380 ms
     MPLS Label=20072  Exp=0  TTL=1  S=1
2    2001:db8:0:75::     .434 ms   .657 ms   .844 ms
```

For the MPLS transport, it displays the ingress labels in each hop.

6.6. LAG

6.6.1. LAG Overview

A link aggregation group (LAG) combines multiple physical links into a single logical interface which is referred to as a bundle interface. These physical links are connected between two devices. The device uses LACP protocol to bundle the member links and create high speed connections. Although a bundle can be created based on static configuration, bandwidth can be increased by adding member links to the bundle. This also allows load sharing among the physical links. Thus, a group of ports combined together is called a link aggregation group, or

LAG.

The LAG interface combines the bandwidth of the individual member links. The properties like speed and bandwidth of the individual member links should be the same to make it part of that LAG. The traffic which is directed towards the LAG interface is sent on the individual member links. This traffic is not pinned to a specific member link but rather determined by a specific flow. This hash could be calculated based on various fields in the packet.

LAG Interface Modes

The LAG interface could be formed statically or dynamically. LACP protocol helps to bring up the interface dynamically. The two modes of LAG interface are:

1. **Static LAG**: In this mode, the member links do not initiate or process any of the LACP packets received. The device brings up the LAG interface without LACP negotiation.
2. **Dynamic LAG**: In this mode, the member links process the LACP packets received. Under this mode, there two sub modes:
   a. **active**: LACP packets are generated on each of the member links on the transmit side.
   b. **passive**: LACP packets are generated on the member link in response to the LACP packet received. That means, at least one side of the LAG should be configured as active to bring up the LAG interface.

Layer2 and Layer 3 Interfaces

LAG interfaces can be used as layer 2 and layer 3 interfaces. A regular layer 2 or layer 3 interface can be created on top of the single LAG interface. These interfaces can be divided based on 802.1q VLAN ID's. Multiple layer 3 interfaces can be created and each of them can be associated with different instances.

LACP (Link Aggregation Control Protocol)

LACP is part of an IEEE specification (802.3ad) that allows several physical ports to be grouped to form a single logical interface. LACP allows a switch to negotiate an LAG by sending LACP packets on its member links. It negotiates the various configuration parameters to bring up the individual member links.

Interface Holddown

The Interface Holddown feature remains supported when interfaces are grouped in a LAG. For more information about the Interface Holddown feature, see Interface Holddown.

Guidelines and Limitations

- You cannot configure logical interfaces on a LAG member ports.
- You cannot configure L2X on a LAG member port.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Supported Number of LAG Interfaces on Platforms

For information on the maximum number of LAG interfaces and members per LAG supported on each hardware platform, refer to the Supported Platforms section of the *Platform Guide*.

6.6.2. LAG Configuration

Creating LAG Interfaces

Syntax:

```
set link-aggregation interface <name> <attribute> <value>
```

Attribute	Description
<name>	Specifies the name of the LAG interface. The supported LAG interface names: 'lag-1' to 'lag-99'.
<description>	Link aggregation interface description

Attribute	Description
mode <mode>	Specifies the LAG mode. The default mode is LACP. The possible modes are: • lacp - In this mode, the member links processes LACP packets received. When you create a LAG interface in LACP mode, the LACP PDUs are sent and received through member interfaces. • static: In this mode, the member links do not initiate or process any of the LACP packets received.
<minimum-link-count>	Specify the minimum number of active member links required for the link aggregation interface.
<member-interface>	Specify name of the member interface.
redundancy-session-id	Specify the value for the redundancy group session ID. Range from 1 to 65535 is allowed.
system-id	Specify the MAC address (as system ID) of the device for the link-aggregation interface.

redundancy-session-id, and system-id attributes can only be used when you deploy RBFS in redundancy mode. For information about LAG configuration when deploying RBFS in redundancy mode, see RBFS Redundancy Solution Guide.

Example: LAG Interfaces Configuration

```
supervisor@rtbrick: cfg> show config link-aggregation
{
    "rtbrick-config:link-aggregation": {
      "interface": [
        {
          "interface-name": "lag-3",
          "mode": "lacp",
          "minimum-link-count": 2,
          "member-interface": [
            {
              "member-interface-name": "ifp-0/0/1",
              "lacp-mode": "active"
            },
            {
              "member-interface-name": "ifp-0/0/5",
              "lacp-mode": "active"
            }
            {
              "member-interface-name": "ifp-0/0/5",
              "lacp-mode": "passive"
            }
          ]
        }
      ]
    }
  }
```

Configuring LAG Member Interfaces

You can add member ports to the LAG interface. The command below allows you to bundle multiple physical interfaces with similar properties like speed, MTU, MRU.

Syntax:

set link-aggregation interface <name> **member-interface** <name> <attribute> <value>

Attribute	Description
lacp-mode <mode>	Specifies the LACP mode. The default lacp-mode is active. **active**: LACP packets are generated on each of the member links on the trad, the receive side. **passive**: LACP packets are generated on the member link in response to the LACP packet received at one side of the LAG should be configured as active to bring the LAG interface.

Attribute	Description
lacp-timeout <timeout-value>	Specifies the timeout for the LACP session. A long timeout is 90 seconds, while a short is 3 seconds (default is short). Setting the timeout value will instruct the partner at which interval it should send the updates (30 seconds for long timeout, 1 second for short timeout). Having mismatching timeouts will not break the operation, even though it is not desirable design-wise. This is because in LACP both Actor and Partner negotiate the transmission rate, so the transmitter sends at the receiver’s expected interval.

Configuring QoS on LAG Interface

RBFS supports QoS at physical interface level for LAG. Users can apply QoS profile at physical interface level through which one common QoS classification can be applied for all traffic on that port, irrespective of the destination logical interface.

The following features are supported:

- Classification (IEEE-802.1)
- Remarking (IEEE-802.1)
- Ingress Policing
- Egress Policing

For information about configuring the above features, refer the *HQoS Configuration Guide*.

- You cannot apply QoS class of service on LAG logical interface
- Currently, queuing and scheduling are not supported

Syntax:

```
set interface <physical interface> class-of-service <class-of-service>
```

Attribute	Description
<interface>	Name of the interface
<class-of-service>	Specifies the class of service

Example:

```
supervisor@rtbrick: cfg> set interface lag-11 class-of-service Retail_profile
supervisor@rtbrick: cfg> commit
supervisor@rtbrick: cfg> show config int lag-11
{
    "rtbrick-config:interface": [
      {
        "name": "lag-11",
        "class-of-service": "Retail_profile"
      }
    ]
  }
```

Configuring L2X on LAG Interface

All forms of L2X that are supported on the regular physical interfaces are supported on LAG. The incoming packet is be matched to a specific L2X profile based on the Cross Connect configuration on the specified LAG interface.

The following match conditions are supported on the LAG interface:

- Incoming LAG interface without any VLAN
- Incoming LAG interface with a single VLAN
- Incoming LAG interface with inner and Outer VLAN
- Incoming LAG interface with any single VLAN
- Incoming LAG interface with inner VLAN and any outer VLAN

For information about configuring L2X, see the L2X Configuration Guide.

The following table provides the L2X match action attributes which are supported on LAG interface.

Attribute	Description
nexthop6 <nexthop>	Next-Hop address
match-type <match-type>	Match types with which traffic can be matched.
service-label <service_label>	Service label value. NOTE: Supported MPLS label values are 0 - 1048575. The reserved MPLS label range is 0 - 15. In RBFS, BGP uses the label range 20000 - 100000. It is recommended to assign label values outside of these reserved ranges to avoid conflicts.
ingress-vlan-operation <ingress-vlan-action>	VLAN operation on ingress side outer VLAN
ingress-outer-vlan <vlan-id>	Outer VLAN at ingress side
outgoing_ifp	Outgoing interface
vlan_operation	VLAN operation
outgoing_outer_vlan1	Outgoing outer VLAN

Configuring Holddown on LAG Interface

By default, the Interface Hold-down feature is enabled. You can disable the feature if you want. It is allowed to configure timer value for options such as 'holddown down delay' and 'holddown up delay'.

Syntax:

set interface <interface-name> <attribute> <value>

Attribute	Description
<interface-name>	Name of the lag. Example: lag-1.
holddown <disable|enable>	Enable or disable the holddown timer feature. By default, the feature is enabled.
holddown-down-delay	Configure the down delay time in milliseconds (ms). The allowed values include 0, 100, 200, 500, 1000, 2000, 5000, and 10000. It overrides the default delay of '0' milliseconds.

Attribute	Description
holddown-up-delay	Configure the up delay time milliseconds (ms). The allowed values include 0, 100, 200, 500, 1000, 2000, 5000, and 10000.

Example Commands:

```
set interface lag-1
set interface lag-1 holddown-down-delay 1000
set interface lag-1 holddown-up-delay 5000
```

Example:

```
supervisor@rtbrick.net: cfg> show config  interface lag-1
{
  "rtbrick-config:interface": [
    {
      "name": "lag-1",
      "holddown-down-delay": "1000",
      "holddown-up-delay": "5000"
    }
  ]
}
```

6.6.3. LAG Operational Commands

Show Commands

Viewing LAG Running Configuration

The following command displays the LAG running configuration on the system.

Syntax:

show config link-aggregation

Example: LAG Running Configuration

```
supervisor@dev1: cfg> show config link-aggregation
{
  "rtbrick-config:link-aggregation": {
    "interface": [
      {
        "interface-name": "lag-4",
        "mode": "lacp",
        "minimum-link-count": 4,
        "member-interface": [
          {
            "member-interface-name": "ifp-0/0/1",
            "lacp-mode": "active",
            "lacp-timeout": "long"
          },
          {
            "member-interface-name": "ifp-0/0/4",
            "lacp-mode": "active",
            "lacp-timeout": "long"
          }
        ]
      }
    ]
  }
}
```

Viewing LAG Information

The following command displays the LAG information.

Syntax:

show lag <options>

Option	Description
<interface-name>	Displays information for a specific LAG interface
detail	Displays detailed LAG information
mode <mode>	Displays information for a LAG mode: static or LACP

Example: Viewing LAG Information

```
supervisor@rtbrick: cfg> show lag detail

  Lag interface name: lag-3
  Status:             Up
  Minimum link count: 2
  Mode:               lacp
    Member interface name: ifp-0/0/1
      Actor system id: 04:f8:f8:e9:bc:83
      Actor key: 107
      Partner system id: 04:f8:f8:e9:bf:83
      Partner key: 43
    Member interface name: ifp-0/0/5
      Actor system id: 04:f8:f8:e9:bc:83
      Actor key: 107
      Partner system id: 04:f8:f8:e9:bf:83
      Partner key: 43
```

Viewing LAG QoS Policer Counters

The following command displays the QoS Policer Counters.

Syntax:

show qos policer counter

Example 1: QoS policer counter

```
supervisor@rtbrick: cfg> show qos policer counter
Interface                         Level  Units      Total           Received
Dropped
lag-27                            1      Packets    0               0
0
                                         Bytes      0               0
0
lag-27                            2      Packets    0               0
0
                                         Bytes      0               0
0
lag-27                            3      Packets    0               0
0
                                         Bytes      0               0
0
lag-27                            4      Packets    0               0
0
                                         Bytes      0               0
0
lag-28                            1      Packets    0               0
0
                                         Bytes      0               0
0
lag-28                            2      Packets    0               0
0
                                         Bytes      0               0
0
lag-28                            3      Packets    0               0
0
                                         Bytes      0               0
0
lag-28                            4      Packets    203             0
203
                                         Bytes      18270           0
18270
lag-29                            1      Packets    0               0
0
                                         Bytes      0               0
0
lag-29                            2      Packets    0               0
```

```
0
                                          Bytes      0                0

0
lag-29                              3     Packets    20591812         18850600
1741212
                                          Bytes      21291933608      19491520400

1800413208
lag-29                              4     Packets    0                0
0
                                          Bytes      0                0

0
lag-27-egress                       1     Packets    0                0
0
                                          Bytes      0                0

0
lag-27-egress                       2     Packets    0                0
0
                                          Bytes      0                0

0
lag-27-egress                       3     Packets    2011928          2011928
0
                                          Bytes      2116548256       2116548256

0
lag-27-egress                       4     Packets    180377           180377
0
                                          Bytes      15574914         15574914

0
lag-28-egress                       1     Packets    0                0
0
                                          Bytes      0                0

0
lag-28-egress                       2     Packets    0                0
0
                                          Bytes      0                0

0
lag-28-egress                       3     Packets    2019661          2019661
0
                                          Bytes      2124683372       2124683372

0
lag-28-egress                       4     Packets    178226           178226
0
                                          Bytes      15398532         15398532

0
lag-29-egress                       1     Packets    0                0
0
                                          Bytes      0                0

0
lag-29-egress                       2     Packets    0                0
0
                                          Bytes      0                0

0
lag-29-egress                       3     Packets    1999328          1999328
0
                                          Bytes      2103293056       2103293056

0
lag-29-egress                       4     Packets    183300           183300
0
                                          Bytes      15893128         15893128

0
```

Syntax:

show qos policer counter <lag interface>

Example 2: QoS policer counter for a specified LAG

```
supervisor@rtbrick: cfg> show qos policer counter lag-29
Interface                          Level  Units      Total            Received
Dropped
lag-29                             1      Packets    0                0
0
                                          Bytes      0                0
0
lag-29                             2      Packets    0                0
0
                                          Bytes      0                0
0
lag-29                             3      Packets    21241103         19499891
1741212
                                          Bytes      21963300502      20162887294
1800413208
lag-29                             4      Packets    0                0
0
                                          Bytes      0                0
0
```

Example 3

```
supervisor@rtbrick: cfg> show qos policer counter lag-29-egress
Interface                          Level  Units      Total            Received
Dropped
lag-29-egress                      1      Packets    0                0
0
                                          Bytes      0                0
0
lag-29-egress                      2      Packets    0                0
0
                                          Bytes      0                0
0
lag-29-egress                      3      Packets    2071701          2071701
0
                                          Bytes      2179429452       2179429452
0
lag-29-egress                      4      Packets    187300           187300
0
                                          Bytes      16221128         16221128
0
```

6.7. HQoS

6.7.1. Hierarchical Quality of Service (HQoS) Overview

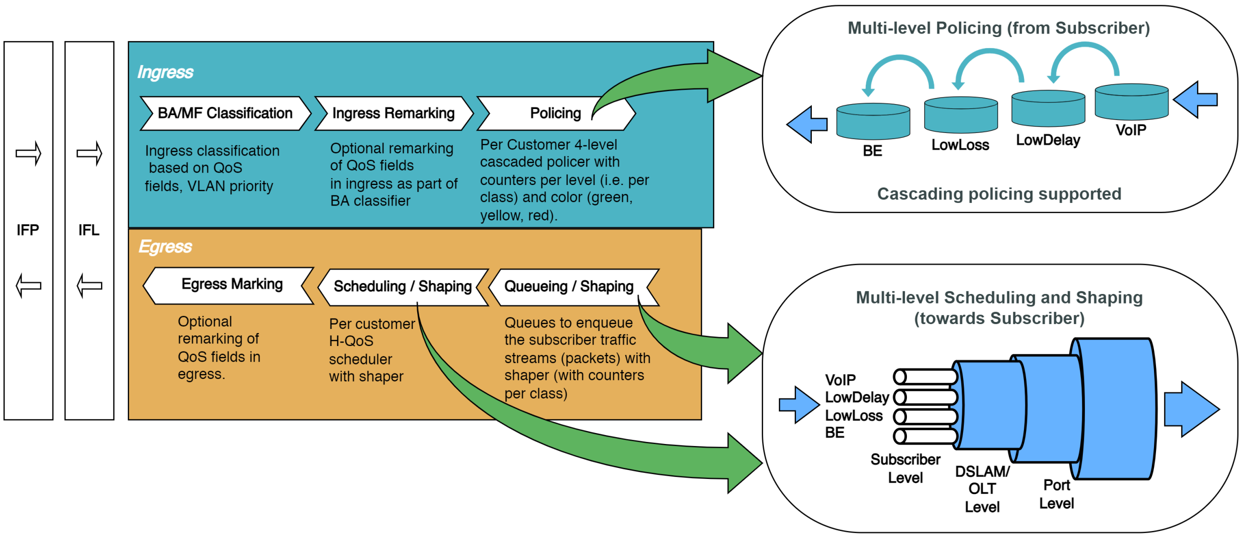
Hierarchical Quality of Service (HQoS) is a traffic classification and prioritization framework that allows you to provide different Service Level Agreements(SLAs) on bandwidth usage. It allocates network resources to services on a prioritized basis. This is achieved by classifying, policing, shaping, scheduling and remarking the traffic based on service types. The HQoS service can be applied to Ethernet, IPv4, IPv6 or MPLS packets. The framework supports QoS at multiple levels. At only one level it allows you to allocate resources between services, but when configured at multiple levels through HQoS, it allows complex prioritization schemes.

The RtBrick Full Stack (RBFS) uses the following HQoS mechanisms:

- **Classifier:** Classifies each incoming packet as belonging to a specific class, based on packet contents. Supported classifiers are Behavior Aggregate (BA) and Multifield (MF). In the BA classifier, packets are classified according to the CoS field: IEEE 802.1p, IPv4/v6 ToS/TC, or MPLS EXP. In the MF classifier, packets are classified using additional fields in the IP header: source IPv4/IPv6 prefix, destination IPv4/IPv6 prefix, L4 source port, L4 destination port, and/or IP protocol.
- **Policer:** Policer is implemented in the ingress to drop the unwanted traffic. Policer supports the Committed Information Rate (CIR), the Committed Burst Size (CBS), the Peak Information Rate (PIR), and the Peak Burst Size (PBS). Drop behavior is to either mark traffic as green, yellow, or drop.
- **Queuing:** Drop unqualified packets in advance using the Weighted Random Early Detection (WRED) technology in the case of congestion to ensure bandwidth for qualified services. This is performed at the egress.
- **Scheduler:** Manage traffic on a device using different algorithms for queue scheduling. Such algorithms include Fair Queuing (FQ), Weighted Round Robin (WRR), and Strict Priority (SP). Queues can be connected to schedulers and schedulers can also be connected to other schedulers.
- **Shaper:** Shaper is implemented in the egress, traffic shaping involves buffering and delaying traffic to shape the flows. Shapers are implemented either on queues or schedulers.
- **Remarking:** Remarking allows you to rewrite the outgoing packet’s codepoint.

Remarking can be performed in the ingress or the egress side of the hardware pipeline.

The figure below shows how QoS does ingress and egress traffic management


IFP
IFL
Ingress
BA/MF Classification
Ingress Remarking
Policing
Ingress classification based on QoS fields, VLAN priority
Optional remarking of QoS fields in ingress as part of BA classifier
Per Customer 4-level cascaded policer with counters per level (i.e. per class) and color (green, yellow, red).
Egress
Egress Marking
Scheduling / Shaping
Queueing / Shaping
Optional remarking of QoS fields in egress.
Per customer H-QoS scheduler with shaper
Queues to enqueue the subscriber traffic streams (packets) with shaper (with counters per class)
Multi-level Policing (from Subscriber)
BE
LowLoss
LowDelay
VoIP
Cascading policing supported
Multi-level Scheduling and Shaping (towards Subscriber)
VoIP
LowDelay
LowLoss
BE
Subscriber Level
DSLAM/ OLT Level
Port Level

The sections below provide a description of various HQoS components that are configured for both ingress and egress traffic.

- QoS Profiles
- Behavior Aggregate (BA) Classifier
- Multifield (MF) Classifier
- Remarking
- Policer
- Class-Policer-Map
- Queueing
- Traffic Class to Queue Mapping
- Queue-Group
- Scheduler
- Priority Propagation
- Shaper
- Scheduler Mapping
- L2TP QoS
- Multi-level H-QoS : Level-1 to Level-5

QoS Profiles

A profile configuration defines the QoS profile that is attached to either a Subscriber interface or an L3 interface.

Profile maps the following QoS constructs to a Subscriber or an L3 interface:

- Behavioral Aggregate (BA) Classifier
- Multifield (MF) Classifier
- Class Policer Map
- Policer
- Class Queue Map
- Scheduler Map
- Remark Map

Behavior Aggregate (BA) Classifier

Classifiers assign the class to which a packet belongs. BA classification is performed on the ingress and maps incoming packet codepoint to a predefined class. BA Classification relies upon markings (that is, codepoint) placed in the headers of incoming packets:

- IEEE 802.1p: Priority - 3 bits
- IPv4: Type of Service byte (ToS) - 8 bits.
- IPv6: Traffic Class (TC) - 8 bits.
- MPLS: Experimental bits (EXP) - 3 bits.

> - IEEE 802.1p and IPv4/IPv6 classifiers are applied on either Subscriber IFL or L3 IFL by attaching the BA classifier to a profile.
> - MPLS Exp classifiers are applied either globally or per instance (to support multiple VPN marking schemes) by attaching the classifier globally or to an instance.

Classifier configuration has the following guidelines and limitations:

- For IPv4: Only ToS-based classification is possible. Currently, DSCP-based classification is not supported.

- For IPv6: TC-based classification is possible.
- For EXP classification, RBFS uses the uniform mode to copy MSB 3-bits from 8-bits IPv4-ToS or IPv6-TC to the EXP field at the time of MPLS encapsulation at the remote box. IPv4/IPv6 Classifiers do not classify labeled traffic, hence MPLS Classifier is required for the same.

- Default class for Queue or Policer is *class-0*. If for an incoming packet's *codepoint* there is no class mapping configured under a classifier, the packet will be classified as *class-0*.
- RBFS supports 8 **classes**: *class-0* to *class-7*.

Ingress Remarking

Ingress remarking is achieved by configuring the "remark-codepoint" field in the Classifier. Ingress remarking rewrites the IPv4-ToS or IPv6-TC field of the incoming packet at the ingress side with configured remark-codepoint. Note that the ingress remarking is not supported for the BA Classifier with match-type MPLS-EXP.

Multifield (MF) Classifier

Multifield (MF) classifiers assign the class to which a packet belongs based on multiple fields. Unlike the BA classifier, where only CoS fields are used for classification, the MF classifier additionally uses the following fields:

- **class**: traffic class of the packet (class-0 to class-7) set by prior BA classifier
- **source prefix**: source IPv4 or IPv6 prefix
- **destination prefix**: destination IPv4 or IPv6 prefix
- **protocol**: UDP or TCP
- **source port**: UDP or TCP source port
- **destination port**: UDP or TCP destination port
- **qos markings**: IPv4 TOS or IPv6 TC header value

The actions supported by a Multifield Classifier are:

- **class**: traffic class to be set (class-0 to class-7)
- **Remark codepoint**: remark codepoint for ingress remarking

RBFS treats all the incoming IPv4-TOS or IPv6-TC QoS field values in the incoming packet as untrusted. So a user is required to set action-remark-codepoint in the MF Classifier configuration to mark the QoS bits in the IP header of the outgoing packet. If action-remark-codepoint is not configured in the MF Classifier, the default value 0 shall be marked in the packet.

The Multifield Classifiers can be bound globally (global.qos.global.config) or via QoS profile (global.qos.profile.config). The global Multifield Classifier applies to all traffic from any instance or interface. The Multifield Classifier assigned via the QoS profile applies only to ingress traffic received on the interface where the profile is bound to it.

The Multifield Classifier is processed after BA classification which allows it to match on selected class from BA classification or to change the assigned class by more granular match conditions. Both classification stages (BA and MF) are optional, they can be combined or used alone, controlled by configuration.

Multifield Classifiers cannot be bound to MPLS core interfaces. Therefore, the downstream traffic (from core to subscriber) should be classified via global Multifield Classifier, while upstream traffic (from subscriber to core) can be classified via Multifield Classifier from the QoS profile, which is instantiated per subscriber with an implicit match on ingress logical interface (InLIF).

- RBFS supports 8 **classes**: *class-0* to *class-7*.
- Per instance MF classifier for MPLS traffic is not supported in RBFS because of hardware limitations.
- The default class for Queue or Policer is **class-0**. If for an incoming packet, there is no MF classification configured, the packet will be classified as *class-0*.
- Priority 1 is reserved for BA Classifier ACL entries, therefore the recommendation is to use Priority starting from 2 for MF Classifier
- If multiple ACL entries are hit in MF having the same priority, the result is unpredictable. So recommendation is to use different priorities for different ACL entries.

Match MPLS traffic

If MF Classifier is to be applied for MPLS traffic (that is, DOWNSTREAM traffic), match MPLS traffic has to be configured in the MF ACL. If not configured, traffic may or may not match the MF ACL entry in the h/w.

Ingress Remarking

Ingress remarking is achieved by configuring the “action remark-codepoint” in the MF Classifier. Ingress remarking rewrites the IPv4-ToS or IPv6-TC field of the incoming packet at the ingress side with configured remark-codepoint.

RADIUS Controlled Dynamic MF Classifier

As described in *RBFS RADIUS Services* document dynamic MF Classifier mapping is supported. The dynamic MF Classifier when configured overrides the MF Classifier mapped via QoS profile for the corresponding subscriber but not other subscribers.

For information about configuring the MF Classifier, see Multifield Classifier (MFC) Configuration.

Layer 2 Access Control Lists (ACL) Priority Bit Egress Classifier

L2 ACL p-bit egress classifier assigns the class to which a packet belongs based on traffic direction (ingress) and outer VLAN priority fields.

- **direction**: Direction of the packet, that is, ingress
- **service-outer-vlan-priority**: Qualifies the packet based on priority bit in the the outer VLAN (IEEE-802.1p)
- **interface-meta-data**: To qualify the packet based on L2 ACL, interface metadata is mandatory (type-1)

The actions supported by a L2 ACL priority bit Egress Classifier are:

- **forward-class**: traffic class to be set on the qualified packets (class-0 to class-7)

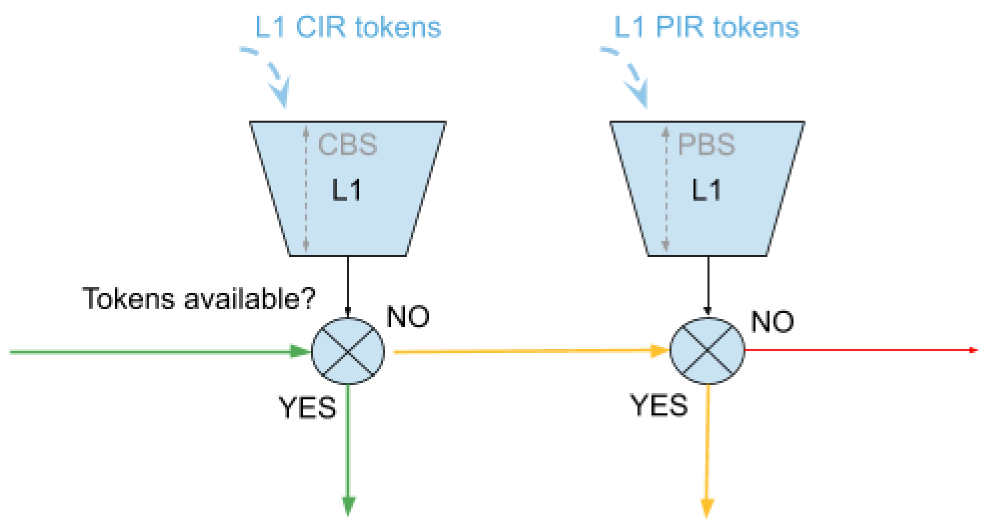
- When EVPN-VPWS L2 interfaces are required to create L2 business services, the shared code points have to be corrected at egress to ensure the correct classification towards the subscriber. The egress classification of packets based on L2

ACL is supported.

Policer

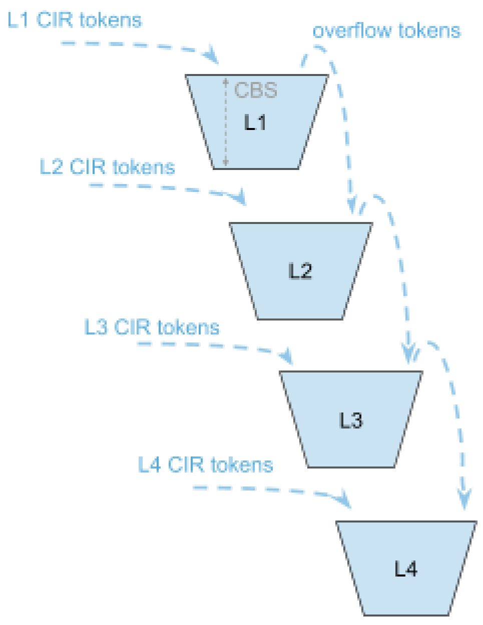
Policer defines the rate at which certain applications can access the hardware resource. So as to rate-limit the traffic from an application, the policer hard-drops the unwanted packets on the ingress side.

In RBFS, policers support the “**two-rate, three-color**” type in a 4-level cascaded mode. This means that each policer level has two rates (CIR and PIR) and three colors (green, yellow and red) with two token buckets as shown below.



This means that traffic below CIR is marked green. Traffic above CIR but below PIR is yellow and above PIR is red. Traffic marked red will be dropped. Traffic marked yellow can be demoted by changing ToS, TC, or EXP using remark map.

In 4-level cascade mode, unused tokens can be passed from higher priority levels to lower priorities where level 1 has the highest and level 4 has the lowest priority as shown in the figure below.



Therefore a lower level configured with CIR 0 can still serve traffic if higher priority levels are not consuming all available tokens.

The available tokens per level are calculated by remaining CIR credits from upper levels and additional credits based on configured CIR per level. Per default, the resulting tokens are not limited. The optional max CIR rate attribute limits tokens from CIR and upper levels. Let us assume levels 1 and 2 are both configured with a CIR of 2Mbps. Without max CIR or max PIR (default behaviour) level 2 can reach up to 4Mbps (level 1 CIR/PIR plus level 2 CIR/PIR). This can be limited by max CIR (for example, 3Mbps). If both level 1 and level 2 have a committed information rate (CIR) of 2Mbps, then level 2 can reach a maximum of 4Mbps (which is the sum of level 1 CIR and level 2 CIR) without any consideration for the maximum CIR. However, the maximum CIR is not relevant for level 1.

In a single-level (levels=1) mode, only level 1 CIR & PIR rates should be configured.

Example

	CIR	RX	TX	PIR	RX	TX
L1	2M	1M	1M	2.5M	1M	1M
L2	3M	20M	9M	3.5M / max CIR 8	20M	8M
L3	4M	20M	0M	4.5M	20M	1M

	CIR	RX	TX	PIR	RX	TX
L4	1M	20M	0M	1.5M	20M	0M
SUM	10M	61M	10M	12M	61M	10M

- Here, M indicates Mbps (Megabits per second)

In columns 2 through 4 of the preceding example table, L1 consumes only 1Mbps of the available 4Mbps and passes the remaining 3Mbps to L2 which adds 6m based on their own configured CIR resulting in 9m.

In columns 5 through 7 of the preceding example table, L1 consumes only 1Mbps of the available 4Mbps and passes the remaining 3m to L2 which adds 6Mbps based on their own configured CIR resulting in 9Mbps. But because of the CIR limit set to 8Mbps, only 8Mbps of 9Mbps can be used at this level. The remaining 1Mbps is now passed to L3 which does not add additional CIR-based credits. In both examples, L4 would be able to reach up to 10Mbps if upper levels are not consuming credits.

RADIUS Controlled Dynamic Policer

The RBFS RADIUS services support dynamic policer rate updates. The dynamic policer rate when configured affects only the QoS instance of the corresponding subscriber but not other subscribers.

Class-Policer-Map

Since RBFS supports up to 8 classes but only four policer levels, there is a need to map multiple classes to the same policer level. A *class-policer-map* defines such mappings. Using class-policer-map configuration, one can map any class to any supported policer level (that includes mapping multiple or all classes to the same level). Similar to a policer, a class-policer-map is attached to a profile.

If class-to-level mapping is not configured, no policing will be applied to the traffic for that class.

7. Queueing

Queuing helps to drop unwanted traffic in advance at the ingress side in case of congestion. This is to ensure bandwidth for qualified services.

RBFS supports the following queueing techniques:

- Tail Drop (TD): This is a conventional congestion avoidance technique. When the network is congested, drop subsequent packets from the queue.
- Weighted Random Early Detection (WRED): This technique requires configuring “Minimum Threshold”, “Maximum Threshold” and “Drop Probability”, which define the start and end range where packets may get discarded. When the average queue size is below the minimum threshold, no packets will be discarded. The drop_probability parameter can be used to specify the drop probability at the max threshold. When the average queue size is between the min and max threshold, the drop probability increases linearly from zero percent (at the min threshold) to drop_probability percent (at the max threshold). When the average queue size is greater than the maximum threshold, all packets are discarded.

    When the average queue size is less than the “Minimum Threshold”, no packets will be discarded.

    When the average queue size is greater than the “Maximum Threshold”, all packets are discarded.

    When the average queue size is between “Minimum Threshold” and “Maximum Threshold”, the drop probability increases linearly from zero percent (at the minimum threshold) to drop probability (at the max threshold).

- Default queue within a queue group is the one mapped to *class-0*. If the classification is not configured for an incoming packet’s codepoint, the packet will be classified as *class-0*. This will be mapped to queue mapped to *class-0* in *Class-Queue-Map*. For more information, see Traffic Class to Queue Mapping.
- Maximum supported Queue size depends upon DRAM/OCB memory. Since OCB is external memory, hardware does not limit the size that can be configured per Queues.

RADIUS Controlled Dynamic Queue

As described for *RBFS RADIUS Services* document dynamic Queue buffer size updates are supported. The dynamic Queue buffer values when configured affect only the QoS instance of the corresponding subscriber but not other subscribers.

Traffic Class to Queue Mapping

The traffic class to queue mapping defines the mapping of classes and queues. The Traffic Class to Queue Mapping is attached to a profile.

- You cannot map two classes to the same queue. The class to queue mapping is 1:1.
- If a queue group is created with four queues, only class-0 to class-3 can be mapped to the queues in class-queue-map; that is, class-4 to class-7 cannot be used.

8. Queue-Group

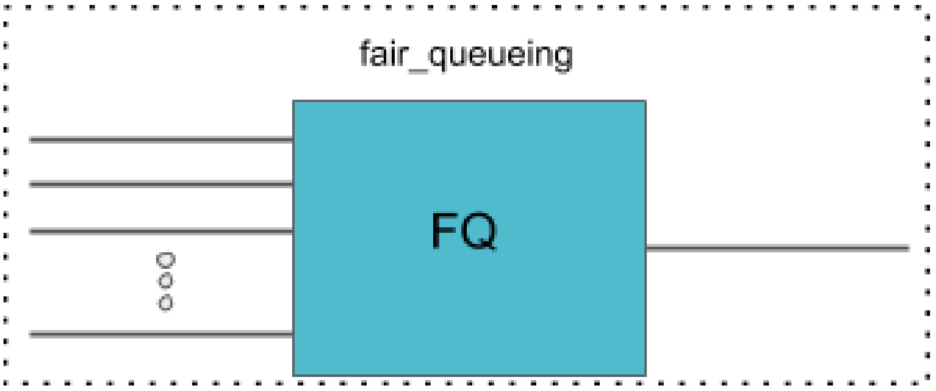
A Queue Group defines the Queue bundle. A Queue Group contains bundle of either 1 or 4 or 8 queues.

9. Scheduler

A scheduler configuration defines scheduler parameters such as type and shaping rate. The shaping rate defined for a scheduler applies to queue(s) associated with it. NOTE: For 1 Queue in Queue-Group, scheduler is not applicable.

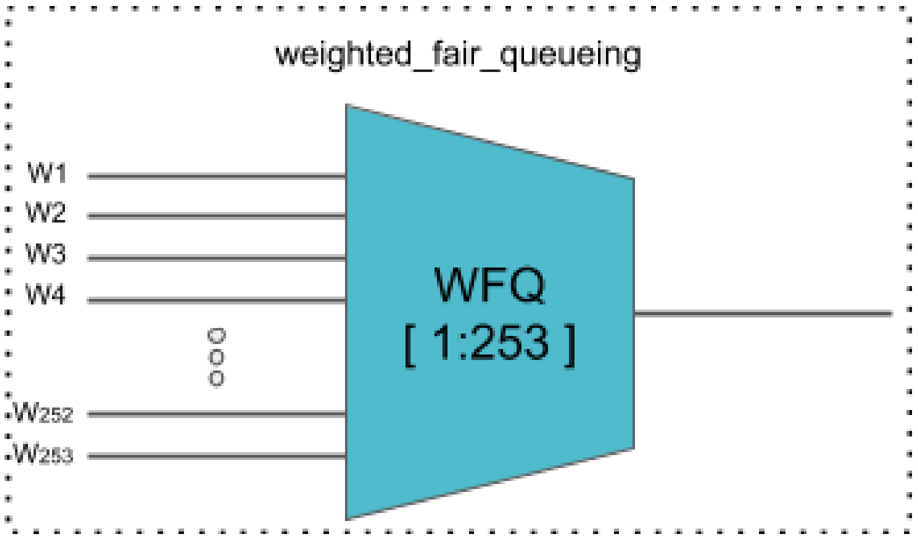
The following scheduler types are supported:

- **Fair Queueing (FQ)**: Uses a round-robin approach to select the next packet to service. This method ensures that all the flows are serviced equally. Configure scheduler type as *fair_queueing* to create FQ scheduler.



- **Weighted Fair Queueing (WFQ)**: Uses a round-robin approach but with no guarantee of flow being serviced equally (like in FQ). The rotation of the next packet to service is based on the weight that is assigned to each flow. Configure scheduler type as *weighted_fair_queueing* to create WFQ scheduler.

  Supported weight: 1 to 253



In any WFQ scheduler, the lower the weight, the higher the bandwidth portion is awarded.

- **Strict Priority (SP)**: Uses priority-based approach to service the flow. SP schedulers are supported in “hybrid” mode only. Hybrid mode combines FQ-WFQ schedulers using strict priority.

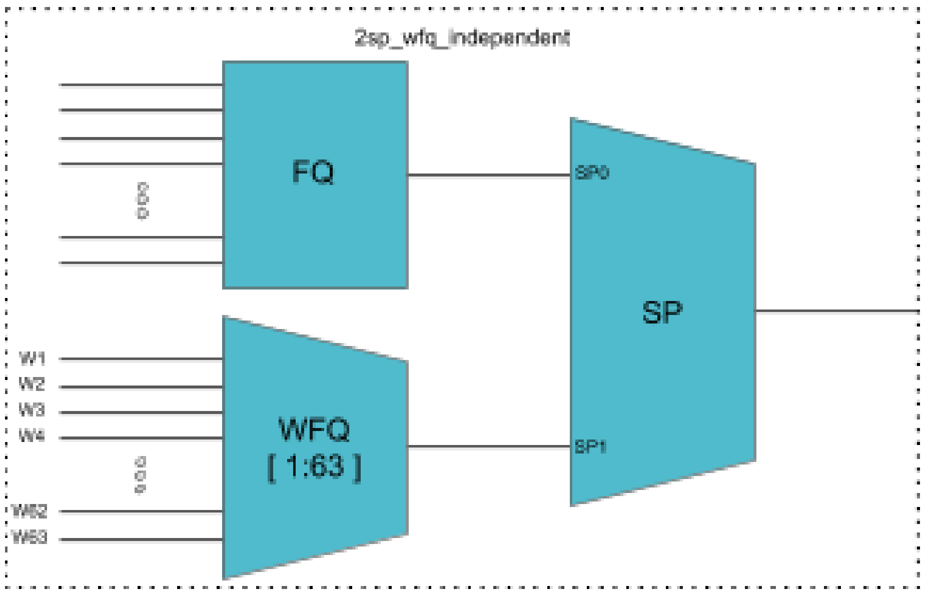
> The priority order for SP is: **strict_priority_0 > strict_priority_1 > strict_priority_2 > strict_priority_3** (where **strict_priority_0** being highest priority and **strict_priority_3** being lowest)

The following SP scheduler types are supported:

- **2 Strict Priority (2SP)**: Uses SP between 1-FQ and 1-WFQ. There are the following types of 2SP hybrid schedulers:

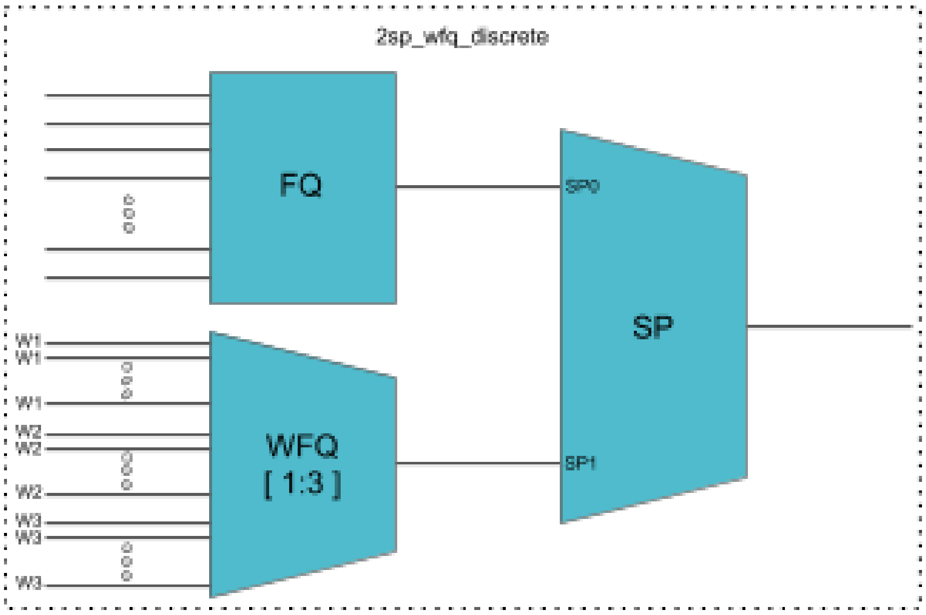
  type **“2sp_wfq_independent”**

  Supported weight: 1 to 63



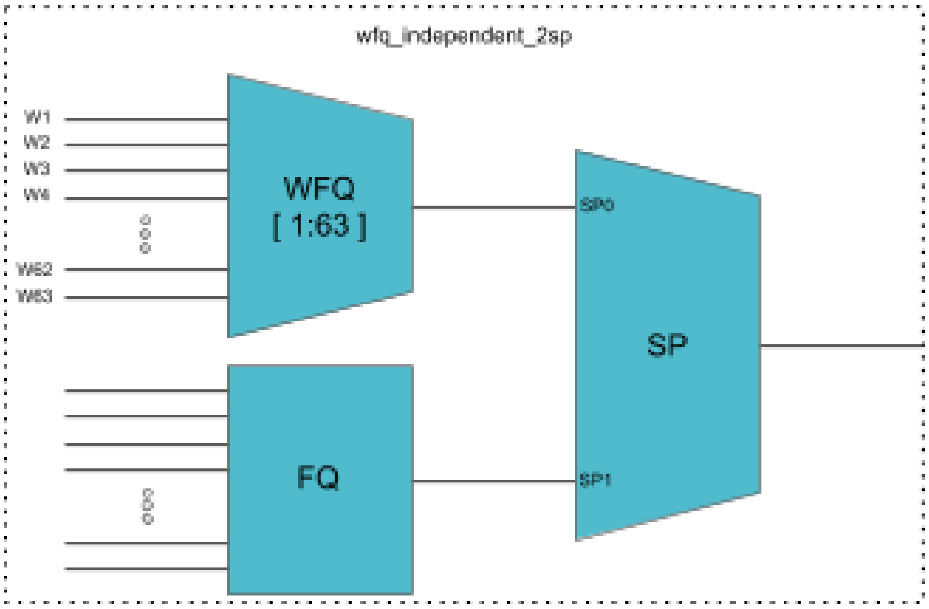
- type **“2sp_wfq_discrete”**

  Supported weight: { 1, 2, 3 }



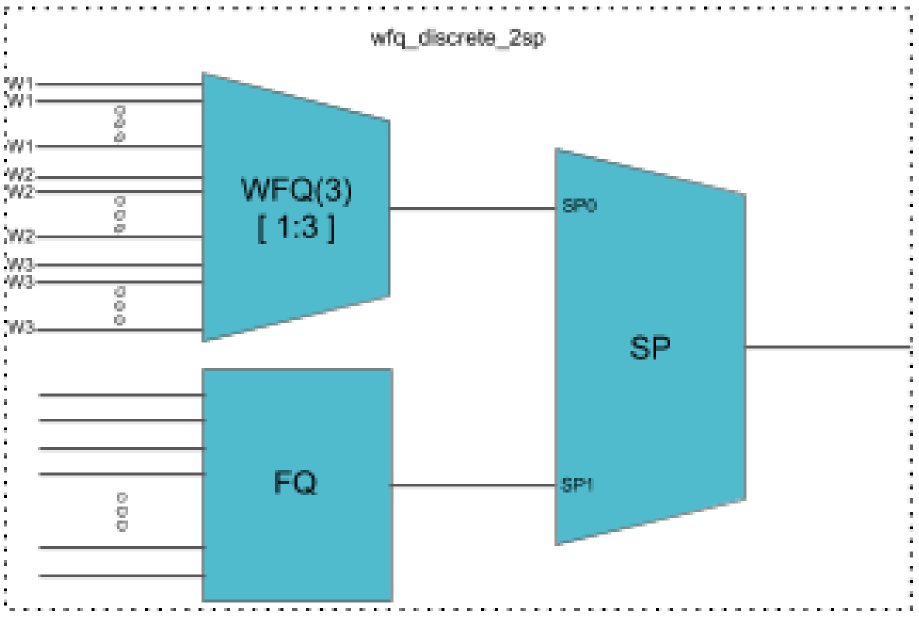
- type **“wfq_independent_2sp”**

  Supported weight: 1 to 63



- type **“wfq_discrete_2sp”**

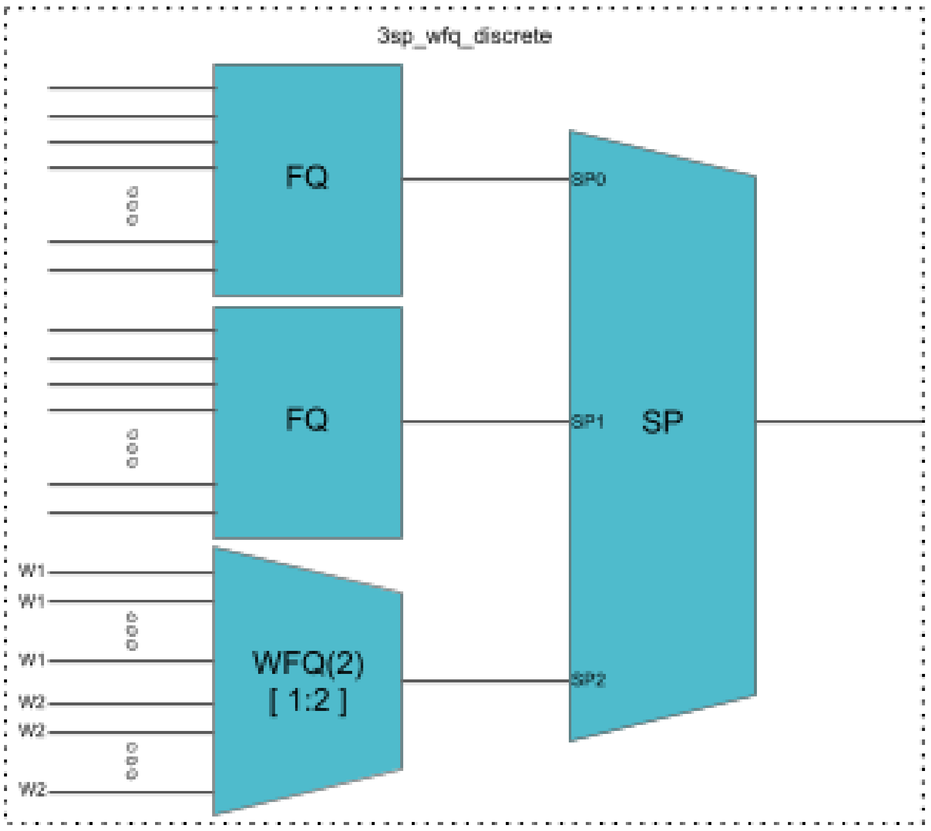
  Supported weight: { 1, 2, 3 }



- **3 Strict Priority (3SP)**: maps 2-FQs and 1-WFQ

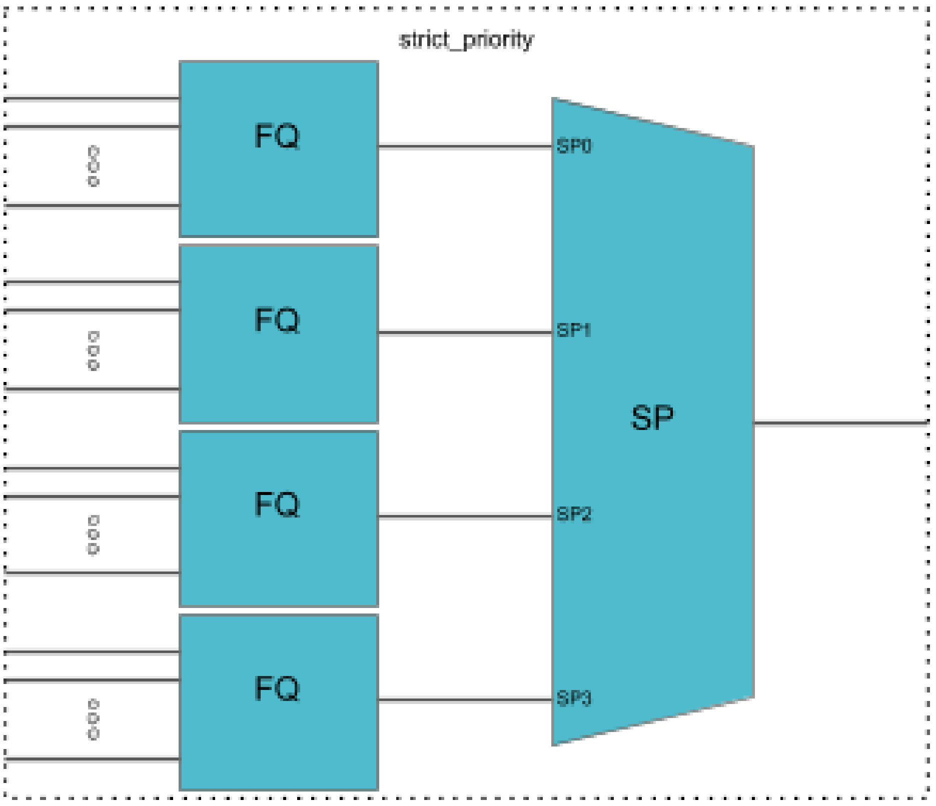
  type: **“3sp_wfq_discrete“**

  Supported weight: { 1, 2 }



- **4 Strict Priority (4SP)**: maps 4-FQs using SP

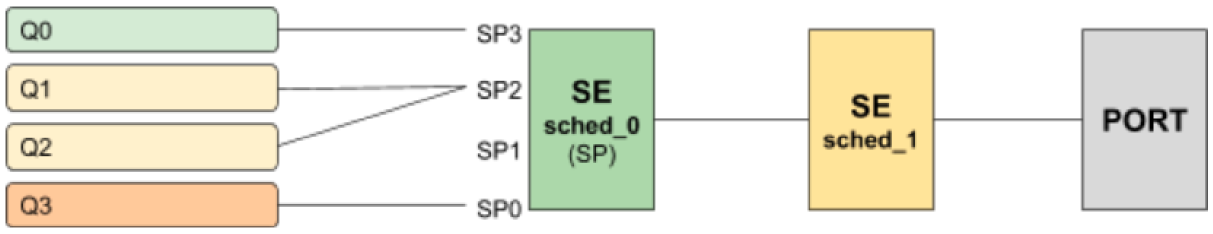
  type **“strict_priority"**



Priority Propagation

Hierarchical QOS (HQoS) on RBFS is implemented by connecting or chaining queues to scheduler elements (Q—> SE), scheduler elements to each other (SE—> SE), and scheduler elements to ports (SE—> PORT). Each scheduler element can have different child connection points based on types described in the section Scheduler.

This means that sched_0 in the example below is not scheduling between the attached queues, but between the different child connection points SP0 to SP3. The scheduler element sched_0 cannot differentiate between Q1 and Q2 in this example because both are connected to SP2.

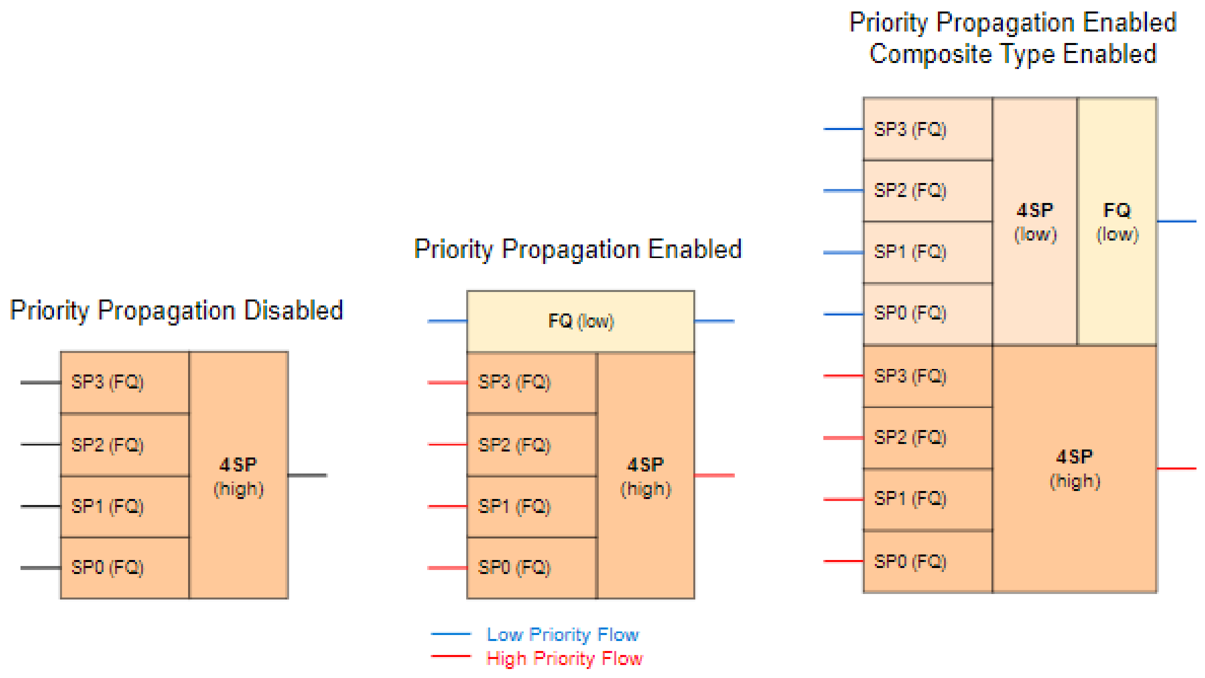


Without priority propagation, each scheduler element can have multiple child connection points but just one parent connection point. Therefore traffic leaving a scheduler element cannot be differentiated by the parent scheduling element. The parent scheduler element sched_1 receives the traffic from sched_0 on the selected child connection point. As already mentioned scheduling within a scheduler element happens between child connection points. Second, a scheduler element has only one parent connection point which can be connected to a child connection point of another scheduler element (output of sched_0   input of sched_1). This results in the situation that all traffic from this SE is handled equally regardless of the queue. This may lead to dropped priority traffic like voice or control traffic in case of congestion in parent elements. For example, if sched_1 has a shaping rate lower than the one of sched_0, it will drop traffic unaware of its original priority.

This problem is addressed with priority propagation which is enabled by default.

With priority propagation, the scheduler elements operate in a dual-flow mode with high and low-priority flows. The credits generated from the physical interface will be consumed by all attached high-priority flows first and only the remaining credits will be available for low-priority flows. In this mode, an implicit FQ element is created for each scheduler element. All queues assigned to low-priority flow will be attached to this element.

An additional composite option of the scheduler element allows also the differentiation between multiple low-priority queues if required. This composite type is created implicitly and does not need to be configured.

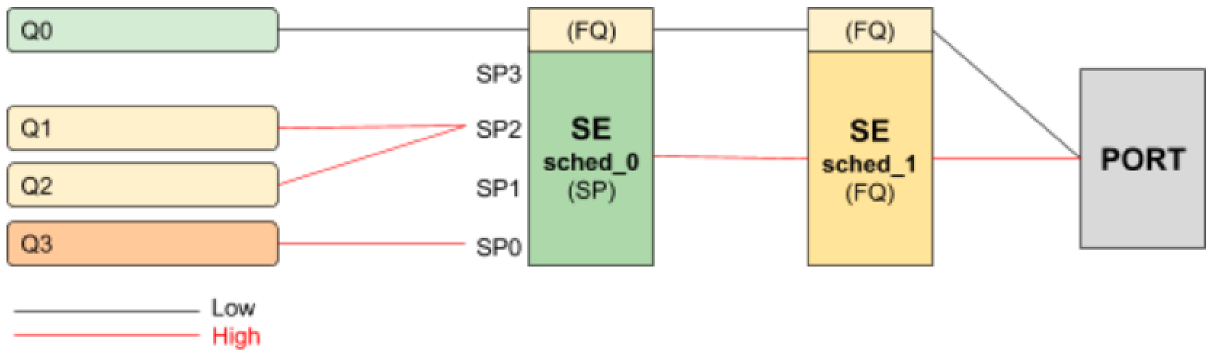


Without priority propagation enabled, each scheduler element consumes only one scheduler resource compared to two elements if enabled. The composite type consumes three scheduler elements.

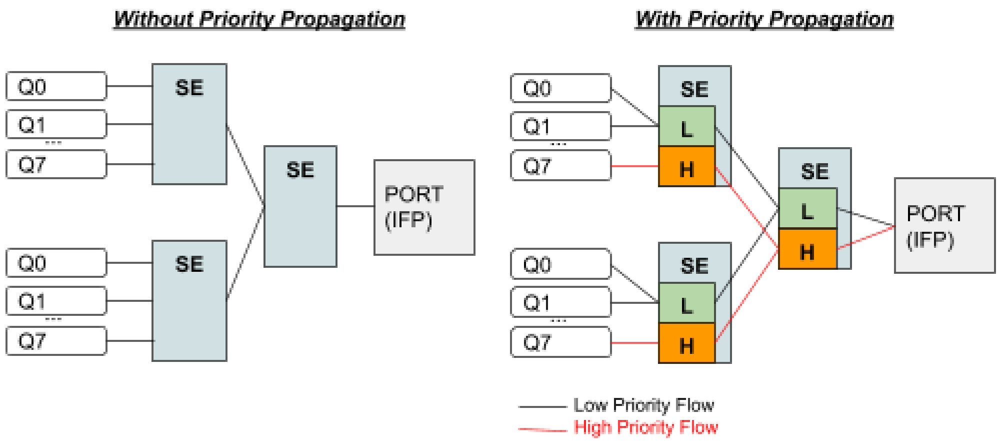
With priority propagation disabled, all traffic is considered a high-priority flow.

Now for each queue, we can select if connected to high-priority or low-priority flow where high-priority flow is selected per default if not explicitly mentioned.

Assuming the example as before but with priority propagation and Q0 assigned to low-priority flow and Q1 - Q3 assigned to high-priority flow.



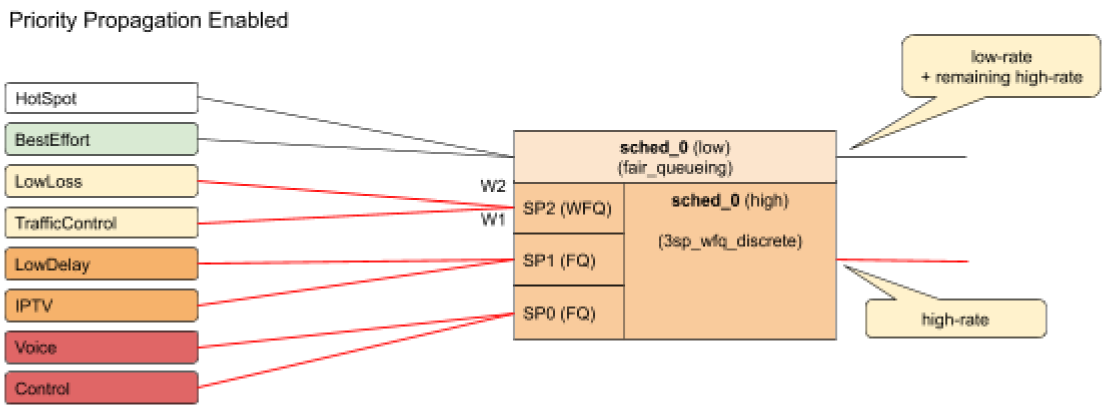
The figure below shows a typical multi-level QoS configuration without priority propagation on the left and with priority propagation on the right side.

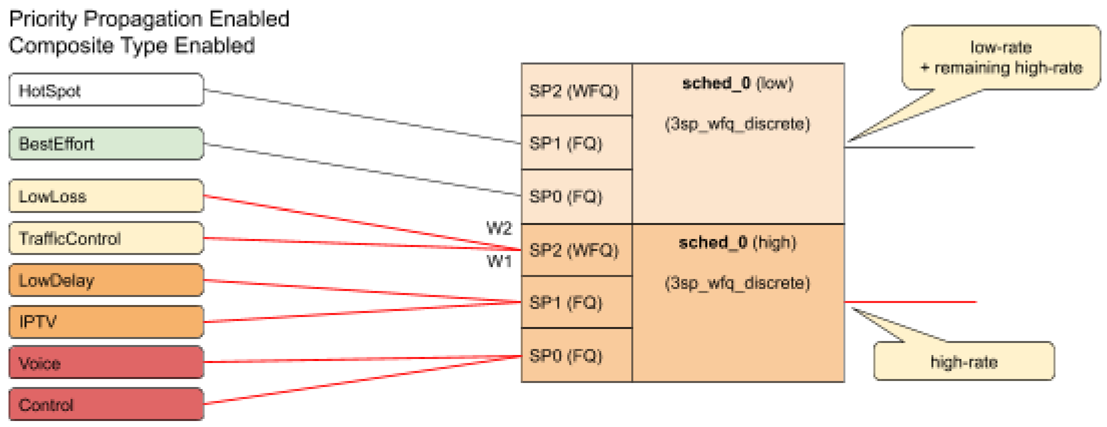


The credits generated from the physical interface will be consumed by high-priority flow first and the remaining credits will be available for low-priority flow. The high-flow traffic at any one element is scheduled based on the type and connection point. Between schedulers, it depends on how they are connected to the parent scheduling element. Per default all levels there is FQ for low and FQ for high-priority flows. The port scheduler is also FQ.

In this mode, each shaper supports two different rates for low and high-priority where the actual shaper rate is the sum of low and high-priority rates. If the low-priority rate is zero, this flow is only served if the high-priority flow is not consuming all credits. An example might be a high rate of 9Mbps and a low rate of 1Mbps which results in 10Mbps for low-priority flow if high-priority flow is not consuming any packets but at least 1Mbps is ensured.

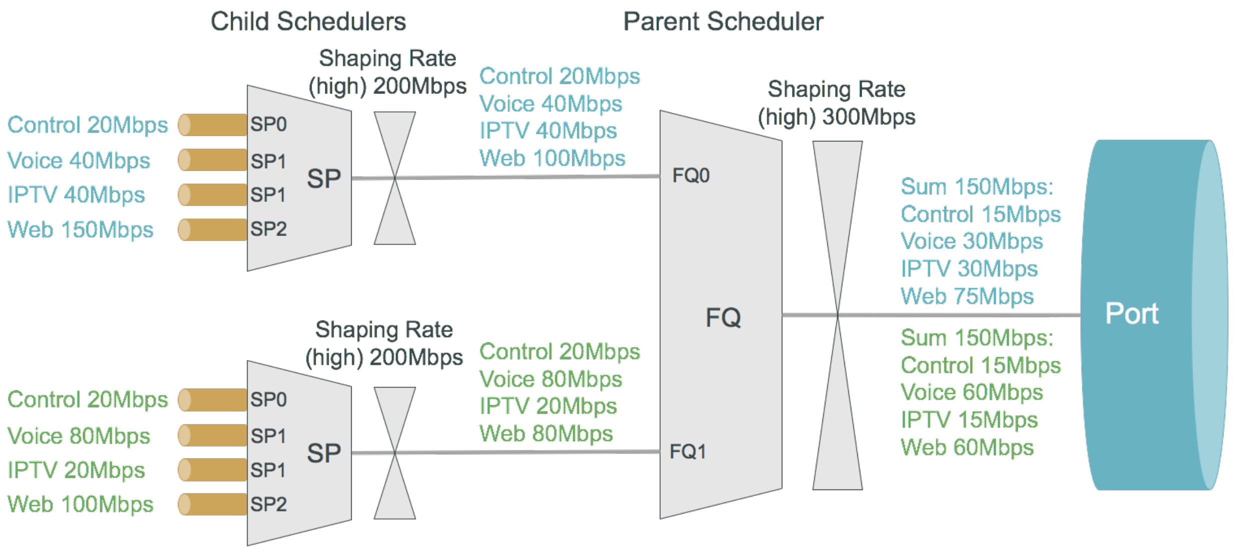
The following example shows a typical access service provider configuration with priority propagation enabled with and without composite type.



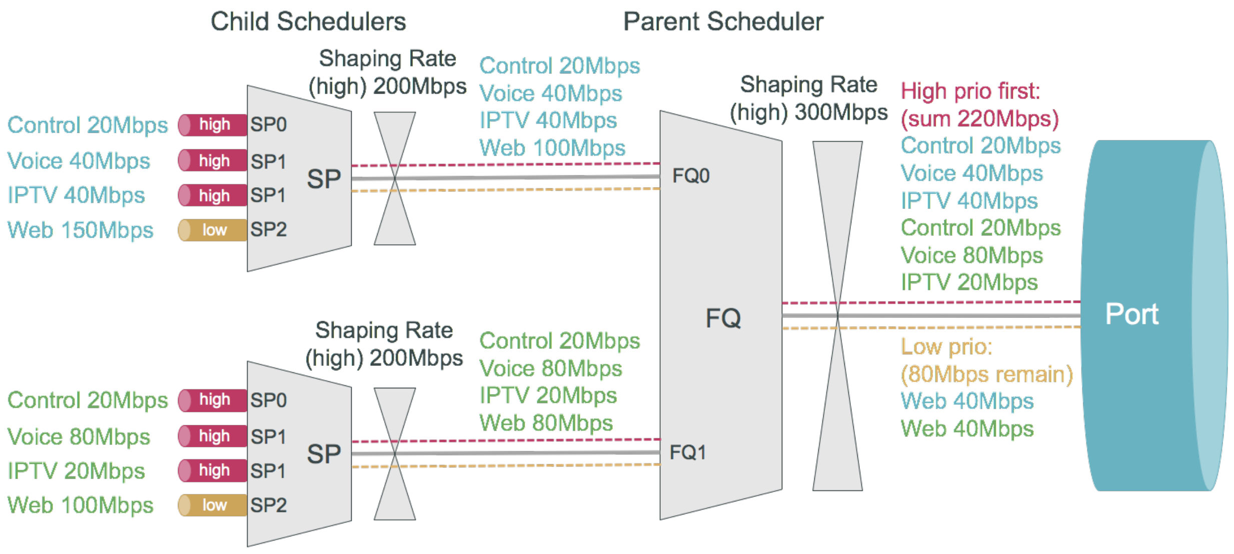


Simple Priority Propagation Scheduling Example

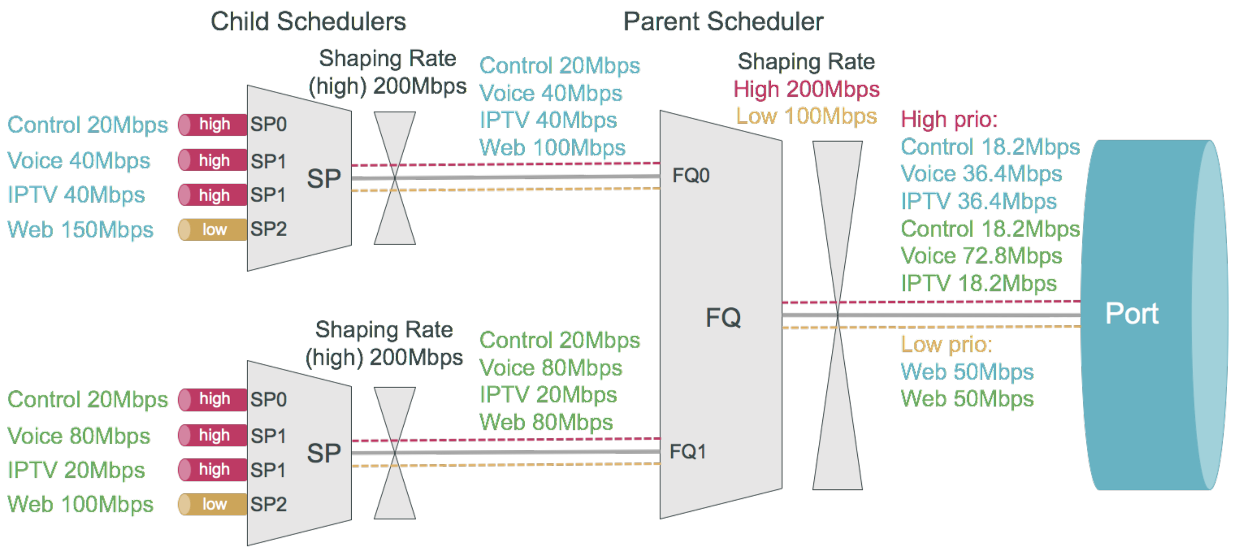
Without priority propagation, the parent scheduler drops traffic equally from all classes as it is unaware of priorities:



With priority propagation, the parent scheduler serves high-priority flows first as shown in the figure below:



With priority propagation and dual-flow shaping, the parent scheduler serves high-priority flows first up to the high-flow shaping rate:


Child Schedulers
Parent Scheduler
Shaping Rate (high) 200Mbps
Control 20Mbps
Voice 40Mbps
IPTV 40Mbps
Web 150Mbps
high
low
SP0
SP1
SP2
SP
Control 20Mbps
Voice 40Mbps
IPTV 40Mbps
Web 100Mbps
Shaping Rate
High 200Mbps
Low 100Mbps
FQ0
FQ
FQ1
High prio:
Control 18.2Mbps
Voice 36.4Mbps
IPTV 36.4Mbps
Control 18.2Mbps
Voice 72.8Mbps
IPTV 18.2Mbps
Low prio:
Web 50Mbps
Web 50Mbps
Port
Shaping Rate (high) 200Mbps
Control 20Mbps
Voice 80Mbps
IPTV 20Mbps
Web 100Mbps
Control 20Mbps
Voice 80Mbps
IPTV 20Mbps
Web 80Mbps

10. Shaper

Shaper is used to rate-limit the traffic at the egress. In RBFS, shapers can be attached to both Queue and Scheduler.

A shaper configuration defines the shaping rate in Kilo-bits-per-seconds (Kbps).

- Setting the shaping rate to 0 (zero) sets the rate to unlimited. Hence it is recommended to configure at least 1 Kbps so that shaping takes place.
- If shaping rates are to be unconfigured, it needs to be done at all scheduler levels.

10.1. Low-rate Shaping

The Low-rate Shaping feature performs queueing and scheduler-level traffic shaping to rates lesser than 1000 Kbps so that the higher-priority (voice) traffic to flow at optimal levels.

Low-rate Shaping is supported only on high-priority flows, that is high-flow configuration parameter.

RBFS Access-Leaf and Multiservice Edge platforms have been enabled with Low-rate Shaping by default. For information about the Low-rate Shaping feature enabled platforms, see section 'Feature/Resource Usage" in the *Platform Guide*.

RADIUS Controlled Dynamic Shapers

RBFS RADIUS services support dynamic shaper updates. The dynamic shaper when configured affects only the QoS instance of the corresponding subscriber but not other subscribers.

11. Scheduler Mapping

Scheduler Map defines the set of relationships between parents and children in egress scheduling hierarchy. A child in a Scheduler Map configuration can be either Queue or Scheduler. A parent in a Scheduler Map configuration can be either a Port or Scheduler.

For 1 Queue in Queue-Group, scheduler-map is not applicable. **Connection Point and Weight**

Child-queue or child-scheduler in a scheduler map configuration is connected to the parent-scheduler at “**connection point (CP)**“. Connection point configuration also has “**weight**” associated with it if the parent has a WFQ scheduler corresponding to that connection point. The valid connection point value for a child to connect to parent **WFQ/FQ** scheduler is **no_priority** and to connect to parent **SP/Hybrid** scheduler is between **strict_priority_0** to **strict_priority_3** (based on a number of Strict Priority points in parent scheduler).

Connection Types

There are five connection types in a scheduler map entry:

- queue_to_port
- queue_to_scheduler
- scheduler_to_scheduler
- scheduler_map_to_scheduler_map
- scheduler_to_port

- For the **queue_to_port** connection type, the scheduler has no role.

12. Remarking

The packet markers set the codepoint in a packet to a particular value, adding the marked packet to a particular behavior aggregate. When the marker changes the codepoint in a packet, it "remarks" the packet. The codepoint in a packet can be IPv4-ToS, IPv6-TC, MPLS-EXP, or IEEE 802.1p field.

The following remarking options are supported in RBFS:

- IEEE 802.1p : Priority - 3 bits.
- IPv4: Type of Service byte (ToS) - 8 bits.
- IPv6: Traffic Class (TC) - 8 bits.
- MPLS-IPv4: MPLS Experimental bits (EXP) - 3 bits.
- MPLS-IPv6: MPLS Experimental bits (EXP) - 3 bits.

IPv4/v6 and IEEE 802.1p remark-map are applied on an interface - subscriber-ifl or l3ifl using Profile Name.

MPLS-IPv4/v6 remark-map is applied either globally or per instance (to support multiple VPN marking schemes) using Remark-Map Name.

In RBFS, remarking can be performed at the ingress or egress:

- **Ingress remarking** is achieved by configuring the **remark-codepoint** field in the Classifier. Ingress remarking rewrites the IPv4-ToS or IPv6-TC at the ingress side with configured remark-codepoint. The configured remark-codepoint can be modified again at the egress side using remark-map. The ingress remarking is supported for IPv4, IPv6, and IEEE 802.1p BA classifiers. Ingress remarking is supported in MF Classifier as well.
- **Egress remarking** is achieved by configuring the **remark-map**. Remark Map is the mapping of **match-codepoint** and **color** to **remark-codepoint**. Egress remarking helps to remark the IPv4-ToS / IPv6-TC field in the IP header, or to write the EXP field in the MPLS label(s), or to write the IEEE 802.1p field in the VLAN header.

Here *Color* is used to set different *remark-codepoint* for same *match-codepoint* based on color marked by the Policer (i.e. *green* or *yellow*). Color is a mandatory field in remark-map. To set the same *remark-codepoint* for a *match-codepoint* irrespective of color, we have to set color as “*all*”.

IPv4-ToS, IPv6-TC, or MPLS-EXP remarking:

- If the *remark-codepoint* is not configured in the BA Classifier or there is no hit in MF Classifier, match-codepoint in the remark-map is the ToS/TC value of the incoming IP packet.
- If the *remark-codepoint* is configured in the BA Classifier and there is no hit in the MF Classifier, match-codepoint is the same value as the remark-codepoint in the BA Classifier
- Irrespective of the *remark-codepoint* configured in the BA Classifier, if there is a hit in the MF Classifier the *match-codepoint* is the same value as the action remark-codepoint (0 if no action *remark-codepoint* configured) in the MF Classifier.

IEEE 802.1p VLAN remarking:

- Class-to-IP based Remark Map for L2TP UPSTREAM traffic is mapped globally. For more information, see the L2TP QoS section.
- In tunnel termination cases (i.e. Downstream traffic from core to Subscriber) the *remark-codepoint* in the MPLS BA Classifier is of no use. Therefore the *match_codepoint* in remark-map at the egress shall be the ToS/TC value of the incoming IP packet.
- In IP tunnel encapsulation cases (i.e. L2TP Upstream traffic from Subscriber to core) the remark-codepoint in the IPv4-TOS BA Classifier is of no use. Therefore the match_codepoint in class-to-ip remark-map at the egress shall be the Class derived from ingress BA Classifier.
- If no MPLS remarking is configured for the Upstream traffic, EXP bits in the MPLS header are derived from IP header TOS/TC bits using the Uniform MPLS mode.
- For VLAN: Only class to IEEE 802.1p remarking is supported.
- For IPv4: Only ToS based remarking is possible. DSCP-based remarking is not possible.
- For IPv6: Only TC based remarking is possible.

- The VLAN priority remarking support for the platforms is as follows:

  UfiSpace S9600-72XC:

  match-codepoint: Value of the IPv4-ToS/IPv6-TC in the incoming IP packet or the remark-codepoint configured in the BA/MF Classifier

  remark-codepoint: Based on the VLAN priority

  Edgecore AS5916-54XKS:

  match-codepoint: MF or BA classifiers are used to determine class at the ingress

  remark-codepoint: Based on the VLAN priority

  QAX Platform: UfiSpace S9500-22XST and Edgecore CSR320 (AS7316-26XB)

  In QAX platform, IP TOS to VLAN bits do not get directly mapped. The IP ToS is mapped to internal priority in classification, and then from internal priority VLAN priority bit remarking is performed.

- **IPv4/v6 and IEEE 802.1p** remark-map is applied on an interface - subscriber-ifl or l3ifl using Profile Name.
- **MPLS-IPv4/v6** remark-map is applied either globally or per-instance (to support multiple VPN marking schemes) using Remark-Map Name.

L2BSA L2X VLAN Priority Remarking:

The L2BSA L2X VLAN priority remarking support for the platforms is as follows: Only A10NSP switches support VLAN operations, access-leaf is transparent.

- Downstream Traffic (Remote ISP to Subscriber):

  On A10NSP Switch:

  The class derived from the VLAN Pbit Classifier is used to remark MPLS Exp and Outer VLAN IEEE-802.1 Pbit Value, inner VLAN Pbit is transparent.

  On Q2C Access Leaf:

The received Outer VLAN IEEE-802.1 Pbit value is retained, explicit remarking at access-leaf is not supported.

- Upstream Traffic (Subscriber to Remote ISP):

  On Q2C Access Leaf:

  To remark the outgoing packet, global MPLS Exp remark map can be used.

  On A10NSP Switch:

  To remark the outgoing packet, VLAN IEEE-802.1 Pbit remark map can be used.

For more information about L2BSA configurations, see the L2BSA User Guide.

L2VPN VLAN Manipulation QoS Remarking:

- VLAN remarking allows for various VLAN manipulations (push, swap, push-push, swap-swap, swap-push) on the outer VLAN in double-tag or single-tag scenarios.
- VLAN remarking for VLAN manipulation involving POP operations (pop, pop-pop, pop-swap) is not supported.

Header Compensation

Queue Compensation

The rate at which the packets are dequeued from a queue depends on the credit received by that queue. The source of the credit received by a queue is the egress port to which the queue is mapped. When a packet is dequeued, the credit balance is decreased by the packet size. But, the packet size that is used must be adjusted to model the packet size at the egress, rather than its actual size at the ingress queue. Thus the header compensation is used to adjust for the differences in header size between ingress queue and egress port. RBFS supports static header compensation configuration per queue (in bytes).

Port Compensation

Similar to queue header compensation where header compensation is performed at the per-queue level, RBFS supports the following header compensation at the per-port level:

- **Ingress Header Compensation**: In line with the header compensation option that we have per-queue, RBFS supports static header compensation configuration at the ingress to be used by the policing. Header compensation changes the effective size of the packet to compensate for changes in header size (such as the CRC removal) when considering the packet for policing. Unlike queue, RBFS ingress header compensation configuration is per ingress port (in bytes).
- **Egress Header Compensation**: In line with the header compensation option that we have per-queue or per-port at the ingress, RBFS supports static header compensation configuration at the egress. The egress header compensation configuration is per egress port (in bytes).

The supported range for header compensation is -64 to +64 bytes.

L2TP QoS

The Layer 2 Tunneling Protocol (L2TP) QoS for upstream is similar to any other locally terminated subscriber. The QoS Profile is mapped dynamically via RADIUS for the L2TP subscribers.

The L2TP QoS for Downstream requires IPv4-TOS based BA Classifier which is mapped to L2TP Tunnel. The same can be achieved by attaching *l2tp-classifier-name* in *global QoS* configuration.

```
forwarding-options {
    class-of-service {
        global {
            l2tp-classifier-name l2tp-ip;
        }
    }
}
```

For Downstream queueing, there is no change. Queueing is applied using QoS Profile similar to locally terminated Subscribers.

The following features are supported for L2TP QoS.

- Upstream

    BA Classifier: IEEE 802.1p

    Policing

        Policer statistics

Remarking

> - L2TP upstream traffic can be remarked by configuring remark-code-point in the classifier configuration.
> - L2TP downstream traffic can be remarked by profile with the remark configuration. A match-codepoint value will be the class value derived from the l2tp-ip classifier.

- Downstream

    BA Classifier: IPv4-TOS

    Queueing/Scheduling/Shaping

        Queue statistics

    Remark-Map: IEEE 802.1p (Class to VLAN priority remarking)

Guidelines

- To avoid control traffic policing/shaping, the assumption is that the IEEE 802.1p bits in Upstream or IPv4-TOS bits in Downstream will be different for control and data traffic, control traffic is expected to have the highest precedence.

    Upstream classification is based on IEEE 802.1p bits.

    Downstream classification is based on IPv4-TOS bits of the outer IP header.

- PBIT (IEEE-802.1p) classification and policing for the upstream traffic from IPoE subscribers is not supported.
- Mark control plane traffic with codepoints based on protocols and layers (such as routing, signaling, and subscriber-management protocols, layer-2, layer-3) and ensure it is treated with higher priority.
- It is recommended to configure MFC (multi-field classifier) for the upstream classification for subscriber traffic.

13. Multi-level H-QoS : Level-1 to Level-5

A sample 5-level requirement looks like this.

Level-1 (IFP)

Physical Interface Shaper.

Level-2 (PON TREE)

Each PON tree is a TDM based shared medium with typically ~2.5 GBit/s (GPON) shared by up to 32 consumers (ONT or DPU).

Level-3 (DPU)

In case of FTTB there is a single DPU with multiple consumers via G.Fast DSL connected which requires an additional hierarchy. This level is not needed for FTTH or FTTC.

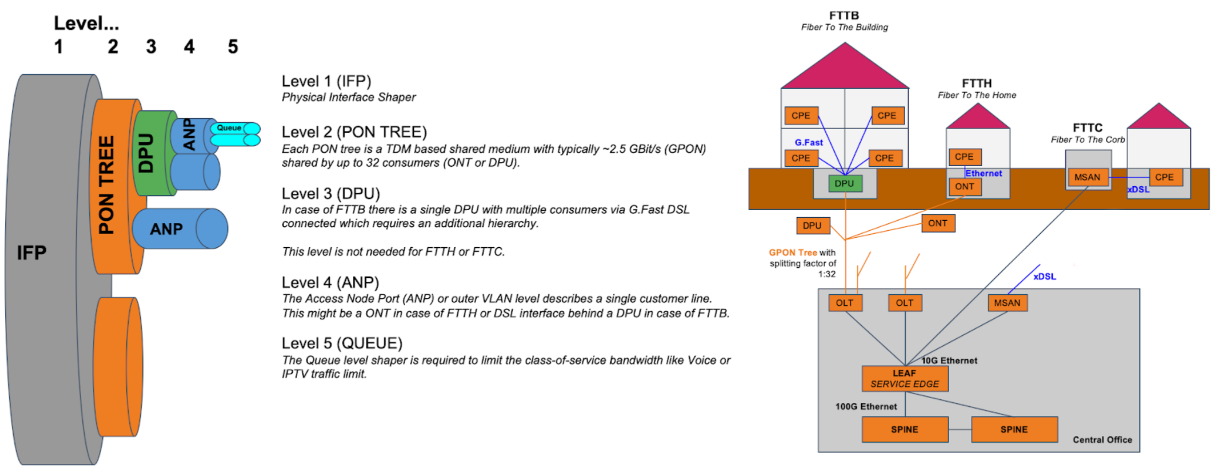
Level-4 (ANP or Session)

The Access Node Port (ANP) or outer VLAN level describes a single customer line. This might be an ONT in case of FTTH or DSL interface behind a DPU in case of FTTB. This level can be also represented on PPPoE sessions as long as just one session is permitted per VLAN.

Level-5 (QUEUE)

The Queue level shaper is required to limit the class-of-service bandwidth like Voice or IPTV traffic limit.

The figure below shows the diagram along with QoS representing Level-1 to Level-5 Hierarchical scheduling.



Level 1 (IFP)
Physical Interface Shaper

Level 2 (PON TREE)
Each PON tree is a TDM based shared medium with typically ~2.5 GBit/s (GPON) shared by up to 32 consumers (ONT or DPU).

Level 3 (DPU)
In case of FTTB there is a single DPU with multiple consumers via G.Fast DSL connected which requires an additional hierarchy.

This level is not needed for FTTH or FTTC.

Level 4 (ANP)
The Access Node Port (ANP) or outer VLAN level describes a single customer line. This might be a ONT in case of FTTH or DSL interface behind a DPU in case of FTTB.

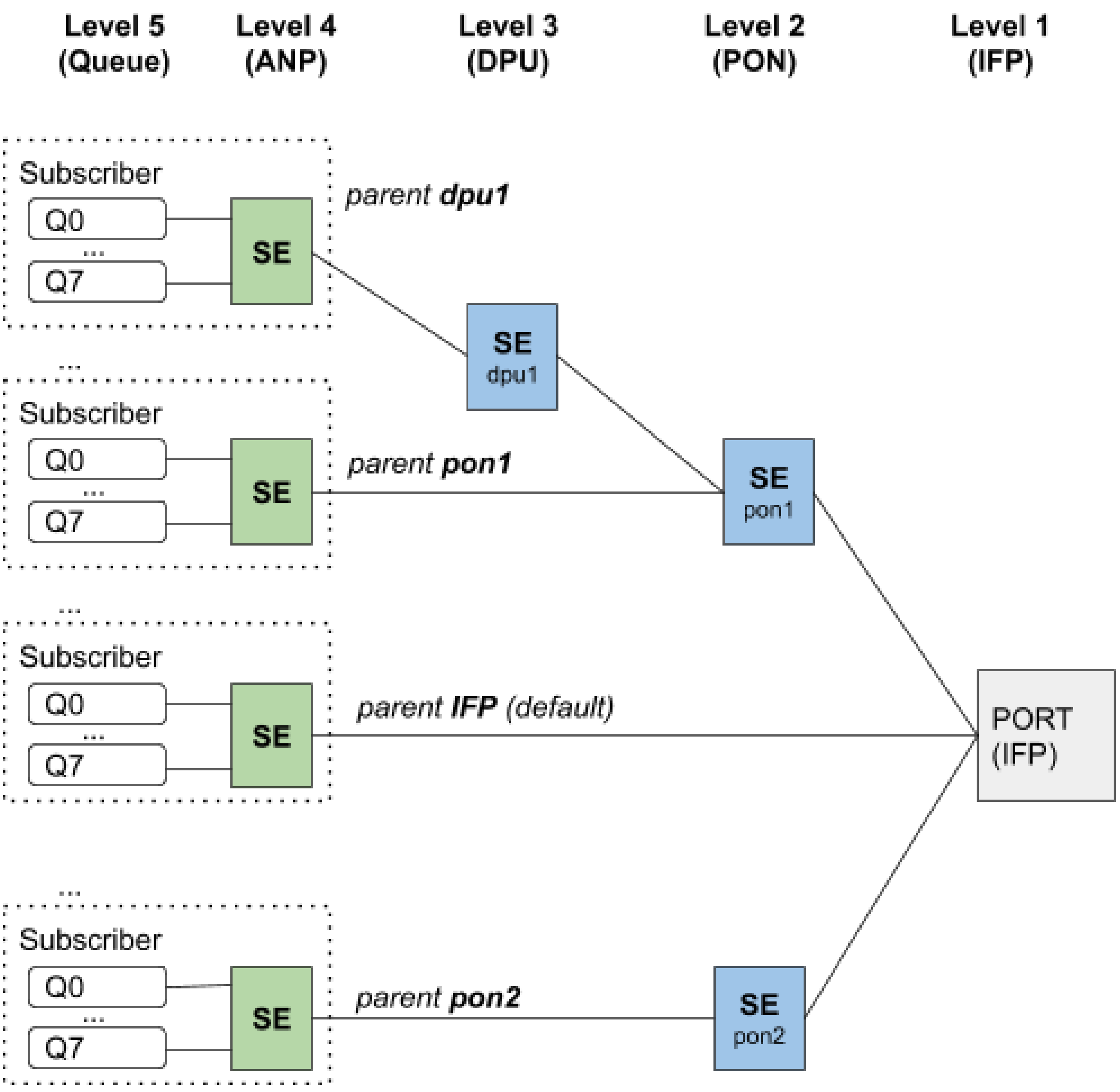
Level 5 (QUEUE)
The Queue level shaper is required to limit the class-of-service bandwidth like Voice or IPTV traffic limit.

Levels 4 and 5 are configured per logical interface (i.e. subscriber-ifl or l3-ifl). Separate scheduler-map representing levels 1 to 3 connectivity shall be statically configured and mapped to the corresponding physical interface (IFP).

The child scheduler in a subscriber scheduler map is connected to the parent scheduler in a physical interface scheduler map using the following way:

- Dynamically via RADIUS in case of dynamic subscribers like PPPoE sessions (Subscriber-IFL).

The figure below shows the same details as the preceding figure before with the different levels but from the DPU-PON-IFP scheduler-map point of view.



QoS Support for L2 and L3 Logical Interfaces

QoS is supported on Layer 2 and Layer 3 logical interfaces.

Support Interface-set for Grouping Logical Interfaces

An interface-set is a collection or group of logical interfaces (that is, L2 or L3) belonging to the same physical interface (IFP).

QoS Support for Interface-set Hierarchy

QoS can be configured at the interface-set hierarchy (service-group), and the IFLs belonging to this interface-set must configure the service-group scheduler to connect to the PON scheduler or directly to the IFP.

Business Service QoS

A business service can be leveraged using an interface-set. A business service comprises of a number of logical interfaces. QoS can be configured at the interface-set hierarchy (service-group) and at the logical interface level as well.

The logical-interface-scheduler is connected to an interface-set (business service group) scheduler and the interface-set-scheduler can be connected to a PON-scheduler or directly to a port-scheduler.

MPLS HQoS

The MPLS HQoS has both UNIFORM and PIPE modes. These modes provide the following functionality:

- During MPLS Encapsulation, MPLS Mode is UNIFORM. MSB 3-bits from 8-bits IPv4-ToS or IPv6-TC are copied to the EXP bits of the newly added MPLS header(s).
- During MPLS Decapsulation, MPLS Mode is PIPE. 8-bit IPv4-ToS or IPv6-TC will be retained and hence it provides ToS/TC codepoint transparency.

For the Uniform MPLS mode mapping between IPv4-ToS or IPv6-TC to MPLS-EXP see the table below:

IPv4-TOS / IPv6-TC	EXP	DSCP
0-31	0	0-7
32-63	1	8-15
64-95	2	16-23
96-127	3	24-31
128-159	4	32-39
160-191	5	40-47
192-223	6	48-55
224-255	7	56-63

Supported Platforms

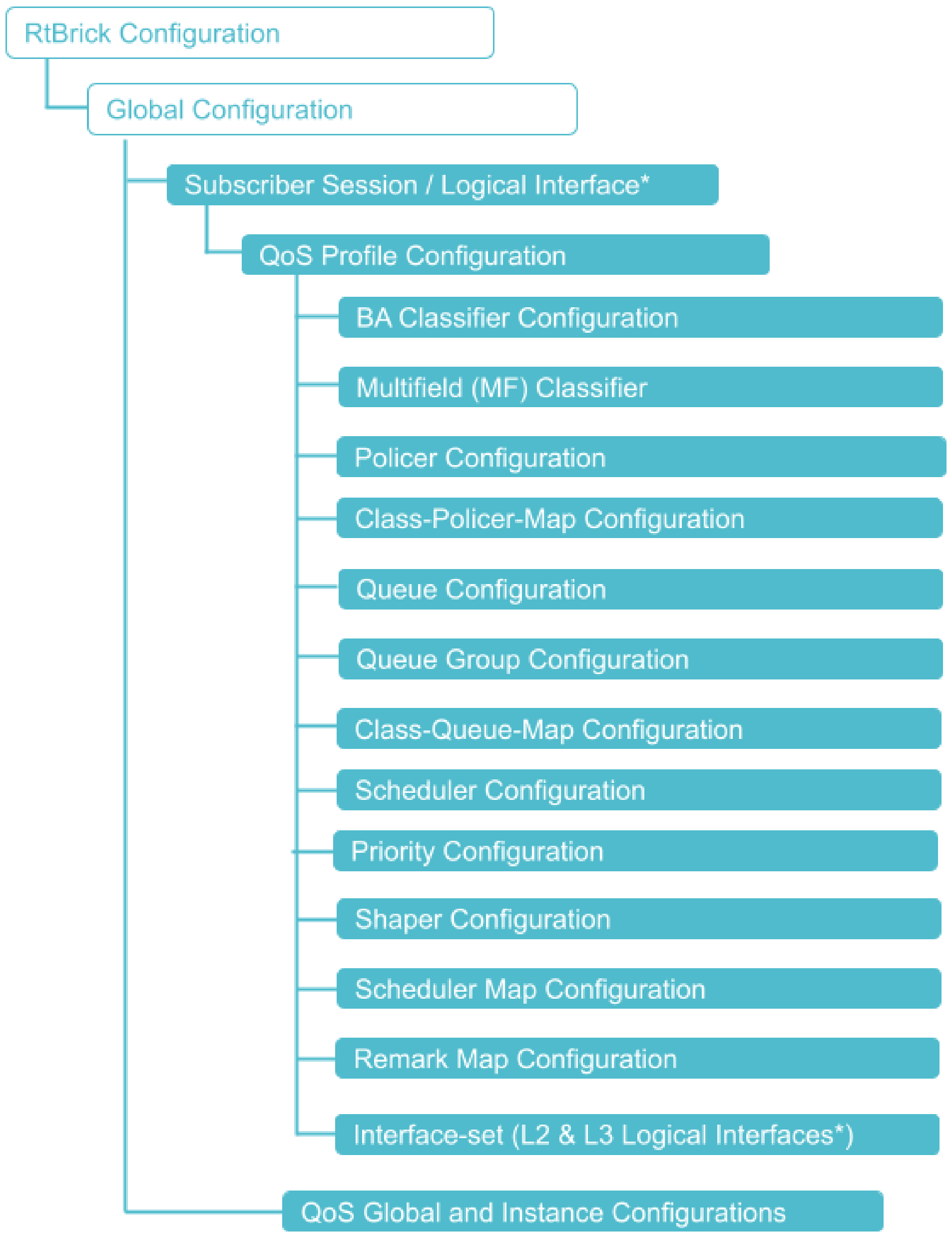
Not all features are necessarily supported on each hardware platform. Refer to the

Platform Guide for the features and the sub-features that are or are not supported by each platform.

13.7. HQoS Configuration

Basic QoS Configuration Elements

The figure below shows the dependencies between the various QoS configuration elements.


RtBrick Configuration
Global Configuration
Subscriber Session / Logical Interface*
QoS Profile Configuration
BA Classifier Configuration
Multifield (MF) Classifier
Policer Configuration
Class-Policer-Map Configuration
Queue Configuration
Queue Group Configuration
Class-Queue-Map Configuration
Scheduler Configuration
Priority Configuration
Shaper Configuration
Scheduler Map Configuration
Remark Map Configuration
Interface-set (L2 & L3 Logical Interfaces*)
QoS Global and Instance Configurations

To configure QoS, perform the following steps which include creating a QoS profile and enabling QoS on a Subscriber-Interface or L3-Interface.

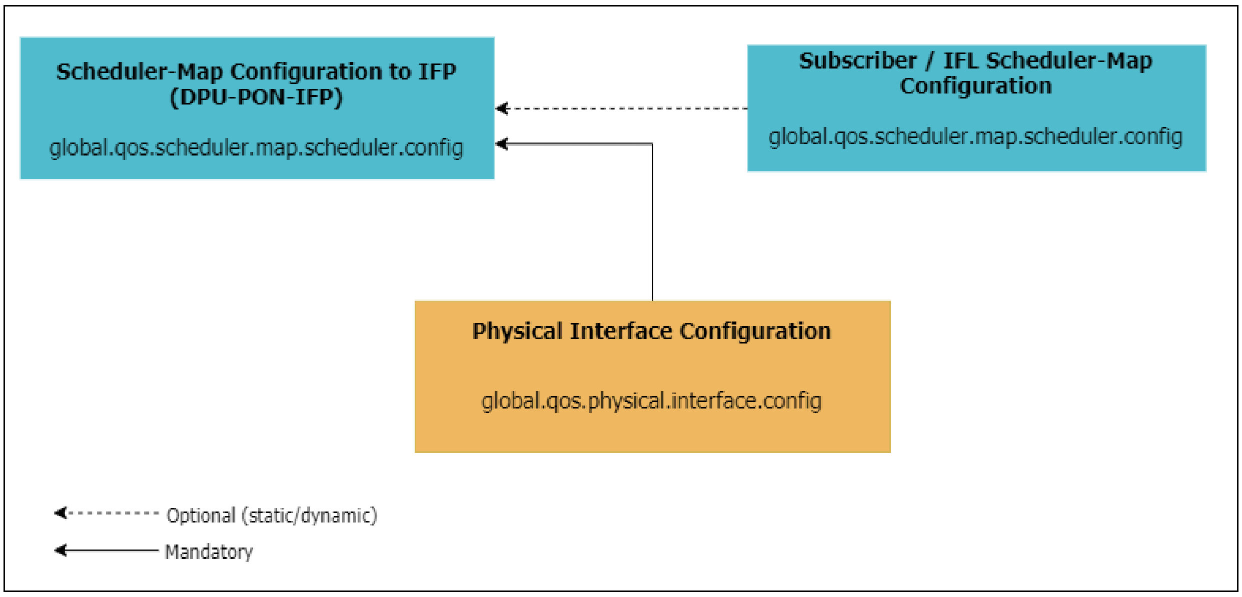
1. Define a QoS profile with classifier, multifield-classifier, class-policer-map, policer, class-queue-map, scheduler-map, and remark-map based on SLAs.
2. Create a Behavioral Aggregate (BA) and/or Multifield (MF) classifier to classify the network traffic at the ingress.
3. Create a policer to police the classified traffic at the ingress with the CIR/PIR defined.
4. Create necessary class-to-policer-map to map the classes to policer levels (mandatory for policing).
5. Create necessary queues with proper size to queue the classified traffic before egressing the traffic.
6. Create queue groups and define queue numbers (1/4/8).
   NOTE: For the system to initialize with a single queue in a Queue-Group, a system reboot is required.
7. Create the necessary class-to-queue map to map the classes to queues (mandatory for queuing).
8. Specify scheduler(s) with type as required.
9. (Optional) Create the shaper (low-flow / high-flow) and attach it to queue(s) and/or scheduler(s).
10. Specify a scheduler map to define a set of relationships between parent (scheduler or port) and child (queue/queue-group or scheduler) at the egress.
11. (optional) Create Remark-Map for QoS field remarking of the outgoing packet.
12. For the downstream traffic, map the MPLS EXP classifier either to an instance or global entity and/or the Multifield (MF) classifier as a global entity (refer to the figure above.)
13. (optional) Map the MPLS-IPv4/IPv6 remark-map either to an instance or configure it as a global entity.
14. For business services QoS, service-group scheduler is connected to PON scheduler or directly to an IFP.
    NOTE: The service group QoS profile is attached to an interface-set.

Priority propagation is enabled by default. To disable the Priority Propagation, we recommend doing this at the beginning and not during an active session.

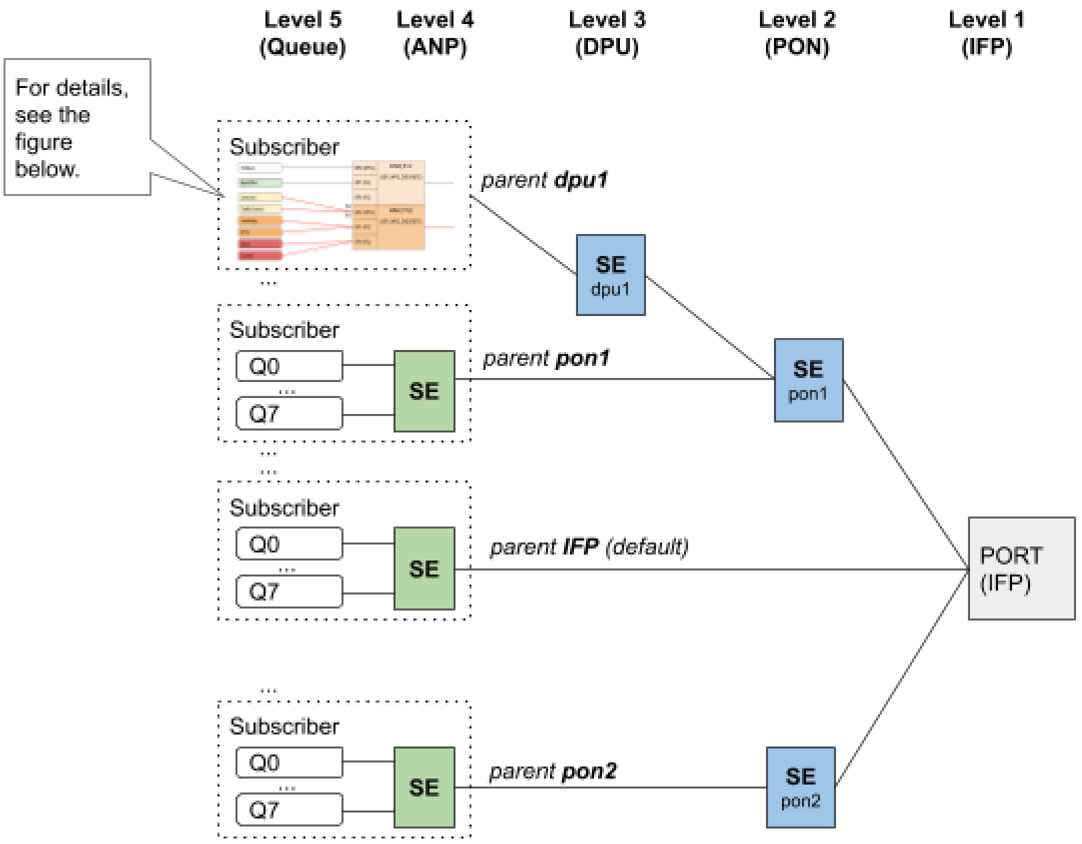
Hierarchical QoS

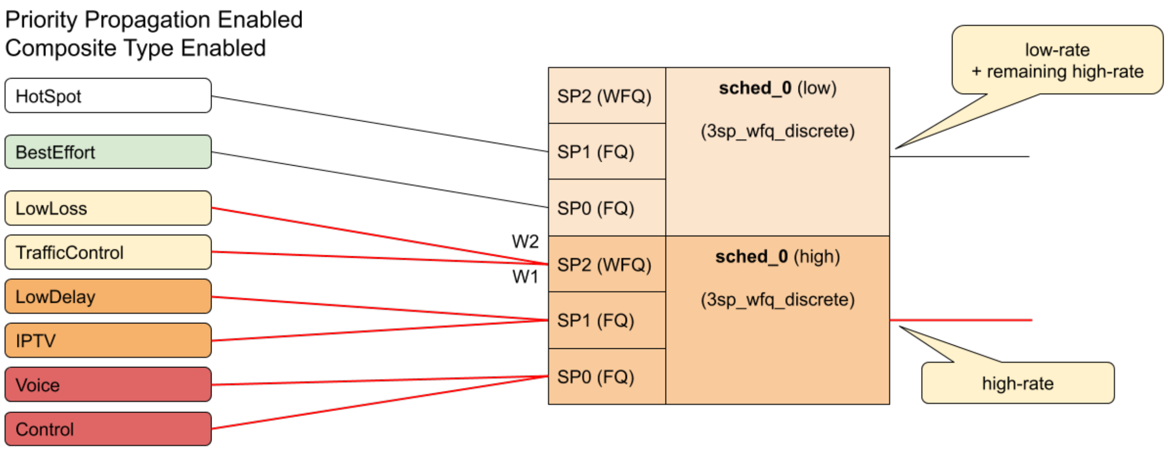
Basic Quality of Service (QoS) is designed to provide a single level of traffic scheduling. In contrast, Hierarchical Quality of Service (HQoS) offers a more sophisticated approach to traffic treatment, utilizing multiple levels of scheduling in a hierarchical manner based on Service Level Agreements (SLAs).

The figure below shows the additional dependencies for Multi-level HQoS.


Scheduler-Map Configuration to IFP (DPU-PON-IFP)
global.qos.scheduler.map.scheduler.config
Subscriber / IFL Scheduler-Map Configuration
global.qos.scheduler.map.scheduler.config
Physical Interface Configuration
global.qos.physical.interface.config
Optional (static/dynamic)
Mandatory

Below are the figures depicting an example of scheduling hierarchy.





To configure HQoS, map the level-3 to level-5 hierarchy in multi-level HQoS to a different scheduler to represent it and map it to the physical interface.

Configuration Syntax and Commands

The configurations in this section exemplify the setup of Hierarchical Quality of Service (HQoS) for a subscriber with a residential profile and a Service Level Agreement (SLA) of 20Mbps upstream and downstream. The setup also encompasses four different service types, namely Best Effort(BE), Low-Delay(LD), Low-Loss(LL) and Voice(VO). Different services are assigned to various policer levels for upstream traffic based on their respective traffic classes. Meanwhile, for downstream traffic, these same services are assigned to different queues depending on their traffic class. The traffic class is determined by the configuration of either the BA or MF classifier.

The following sections describe the HQoS configuration syntax and commands.

- Configuring QoS Profiles
- Behavior Aggregate (BA) Classifier Configuration
- Multifield Classifier (MFC) Configuration
- L2 ACL Priority Bit Egress Classifier Configuration
- Policer Configuration
- Class-Policer-Map Configuration
- Queue Configuration
- Queue-Group Configuration
- Class-Queue-Map Configuration
- Scheduler Configuration
- Shaper Configuration
- Scheduler-Map Configuration
- Remark-Map Configuration

13.8. Configuring QoS Profiles

A profile configuration defines the QoS profile that is installed on the subscriber or L3 interfaces. The following sections explain the different elements of the QoS profiles that you can create and attach.

Use the following CLI syntax to configure a QoS profile:

```
set forwarding-options class-of-service profile <profile-name>
<attribute> <value>
```

Attribute	Description
<profile-name>	Specifies the name of the QoS profile
classifier-name <classifier-name>	Specifies the name of the BA classifier
multifield-classifier-name <multifield-classifier-name>	Specifies the name of the multifield classifier
class-policer-map-name <class-policer-map-name>	Specifies the name of the map that associates a class with a policer level.
policer-name <policer-name>	Specifies the policer name
class-queue-map-name <class-queue-map-name>	Specifies the name of the map that associates a class with a queue
scheduler-map-name <scheduler-map-name>	Specifies the name of the scheduler map
remark-map-name <remark-map-name>	Specifies the name of the remark map
egress-class-policer-map-name <egress-class-policer-map-name>	Specifies the name of the egress Class Policer Map. This configuration applies only to the L2X traffic on the LAG interface.
egress-policer-name <egress-policer-name>	Specifies the name of the egress policer. This configuration applies only to the L2X traffic on the LAG interface.

In the following example, the QoS profile residential is configured with classifier-name subs-pbit-class, class-policer-map policer-map-residential, policer policer-residential, class-queue-map subs-4queues, scheduler-map subs-4queues-residential, and remark-map subs-remarking-residential.

```
set forwarding-options class-of-service profile residential
set forwarding-options class-of-service profile residential classifier-name subs-pbit-class
set forwarding-options class-of-service profile residential class-queue-map-name subs-4queues
set forwarding-options class-of-service profile residential remark-map-name subs-remarking-residential
set forwarding-options class-of-service profile residential policer-name policer-residential
set forwarding-options class-of-service profile residential class-policer-map-name policer-map-residential
set forwarding-options class-of-service profile residential scheduler-map-name subs-4queues-residential
commit
```

The following example shows the QoS profile configuration of the residential profile:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service profile
{
  "rtbrick-config:profile": [
    {
      "profile-name": "residential",
      "classifier-name": "subs-pbit-class",
      "class-queue-map-name": "subs-4queues",
      "remark-map-name": "subs-remarking-residential",
      "policer-name": "policer-residential",
      "class-policer-map-name": "policer-map-residential",
      "scheduler-map-name": "subs-4queues-residential"
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

Behavior Aggregate (BA) Classifier Configuration

Use the following CLI syntax to configure the BA classifier:

set forwarding-options class-of-service classifier <classifier-name> <attribute> <value>

Attribute	Description
<classifier-name>	Specifies the classifier name
match-type <match-type>	Specifies the type of traffic to classify, that is, ipv4-tos, ipv6-tc, ieee-802.1, ieee-802.1-inner, exp
match-type <match-type> codepoint <codepoint>	Specifies the code-point value based on the match-type
match-type <match-type> codepoint <codepoint> class <class>	Specifies the traffic class as class-0, class-1, class-2, class-3, class-4, class-5, class-6, and class-7
match-type <match-type> codepoint <codepoint> remark-codepoint <remark-codepoint>	Specifies the remark-codepoint that is used for remarking

The following example configures the BA classifier subs-pbit-class for traffic ieee-802.1 with a match code point 2 classified as class class-1.

```
set forwarding-options class-of-service classifier subs-pbit-class match-type ieee-802.1 codepoint 2 class
class-1
commit
```

The following example shows BA classifier configuration.

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service classifier
{
  "rtbrick-config:classifier": [
    {
      "classifier-name": "subs-pbit-class",
      "match-type": [
        {
          "match-type": "ieee-802.1",
          "codepoint": [
            {
              "codepoint": 2,
              "class": "class-1"
            }
          ]
        }
      ]
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

The following configuration sets ieee-802.1-inner classifier match-type.

```
set forwarding-options class-of-service classifier subs-inner-pbit
set forwarding-options class-of-service classifier subs-inner-pbit match-type
ieee-802.1-inner
set forwarding-options class-of-service classifier subs-inner-pbit match-type
ieee-802.1-inner codepoint 2
set forwarding-options class-of-service classifier subs-inner-pbit match-type
ieee-802.1-inner codepoint 2 class class-1
```

The following example shows the configuration for the ieee-802.1-inner classifier match-type.

```
supervisor@rtbrick>: cfg> show config forwarding-options class-of-service classifier subs-inner-pbit
{
  "rtbrick-config:classifier": [
    {
      "classifier-name": "subs-inner-pbit",
      "match-type": [
        {
          "match-type": "ieee-802.1-inner",
          "codepoint": [
            {
              "codepoint": 2,
              "class": "class-1"
            }
          ]
        }
      ]
    }
  ]
}
```

Multifield Classifier (MFC) Configuration

Use the following CLI syntax to configure the multifield classifier.

set forwarding-options class-of-service multifield-classifier <attribute> <value>

Starting from Release 20.10.2, Multifield Classifier configuration requires explicit use of ordinal keywords.

Attribute	Description
acl <l3v4 | l3v6> <…>	Specifies the l3v4 / l3v6 ACL rule for multifield classifier configurations. For more information on configuring the ACL match rules, see the sections below.

IPv4 ACL Configuration Configuration

Use the following CLI syntax to configure the IPv4 ACL Match Configuration for the multifield classifier:

set forwarding-options class-of-service multifield-classifier acl l3v4 rule <rule-name> ordinal <ordinal-value> <attribute> <Value>

Attribute	Description
<rule-name>	Specifies the multifield classifier rule name
ordinal <ordinal-value>	Specifies the ordinal that is used for traffic policy rule referencing
match <destination-ipv4-prefix>	Specifies the destination IPv4 prefix address
match <destination-ipv4-prefix-list>	Specifies the destination IPv4 prefix list name

Attribute	Description
match <direction>	Specifies the acl l3 traffic direction match
match <ip-protocol>	Specifies the IP protocol such as UDP or TCP
match <destination-l4-port>	Specifies the Layer 4 destination port number
match <ipv4-tos>	Specifies the IPv4 ToS value
match <ipv4-dscp>	Specifies the IPv4 Differentiated Services Code Point (DSCP) value
match <forward-class>	Specifies the forward class name
match <mpls-traffic>	Specifies the MPLS traffic
match <source-ipv4-prefix>	Specifies the source IPv4 prefix address
match <source-ipv4-prefix-list>	Specifies the source IPv4 prefix list name
match <source-l4-port>	Specifies the Layer 4 source port number
action forward-class <class>	class-0, class-1, class-2, class-3, class-4, class-5, class-6, class-7
action remark-codepoint <remark-codepoint>	Specifies the remark-map codepoint value

Multifield classification commonly matches on IP address fields, the IP protocol type field, or the port number in the UDP or TCP pseudo-header field.

The l3v4 acl-type multifield classifier is configured with a qualifier/match based on ipv4-tos (128, 160) and source-ipv4-prefix (132.1.1.3/32) with action as forward-class (class-0,class-1).

```
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1001
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1001 match
ipv4-tos 128
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1001 match
source-ipv4-prefix 132.1.1.3/32
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1001 action
forward-class class-0
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1002
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1002 match
ipv4-tos 160
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1002 match
source-ipv4-prefix 132.1.1.3/32
set forwarding-options class-of-service multifield-classifier acl l3v4 rule global_mfc ordinal 1002 action
forward-class class-1
```

```
commit
```

The following example shows the IPv4 Match Configuration for the multifield classifier:

```
supervisor@rtbrick>multiservice-edge.rtbrick.net: cfg> show config forwarding-options class-of-service
multifield-classifier acl l3v4 rule global_mfc
{
    "rtbrick-config:rule": [
      {
        "rule-name": "global_mfc",
        "ordinal": [
          {
            "ordinal-value": 1001,
            "match": {
              "ipv4-tos": 128,
              "source-ipv4-prefix": "132.1.1.3/32"
            },
            "action": {
              "forward-class": "class-0"
            }
          },
          {
            "ordinal-value": 1002,
            "match": {
              "ipv4-tos": 160,
              "source-ipv4-prefix": "132.1.1.3/32"
            },
            "action": {
              "forward-class": "class-1"
            }
          }
        ]
      }
    ]
  }
```

IPv6 ACL Configuration Configuration

Use the following CLI syntax to configure the IPv6 ACL match configuration for the multifield classifier:

set forwarding-options class-of-service multifield-classifier acl l3v6 rule <rule-name> ordinal <ordinal-value> <attribute> <value>

Attribute	Description
<rule-name>	Specifies the multifield classifier rule name
ordinal <ordinal-value>	Specifies the ordinal that is used for traffic rule referencing
match <destination-ipv6-prefix>	Specifies the destination IPv6 prefix address

Attribute	Description
match <destination-ipv6-prefix-list>	Specifies the destination IPv6 prefix list name
match <ip-protocol>	Specifies the IP protocol such as UDP or TCP
match <destination-l4-port>	Specifies the Layer 4 destination port number
match <ipv6-tc>	Specifies the IPv6 traffic class value
match <forward-class>	Specifies the forward class name
match <mpls-traffic>	Specifies the MPLS traffic
match <source-ipv6-prefix>	Specifies the source IPv6 prefix address
match <source-ipv6-prefix-list>	Specifies the source IPv6 prefix list name
match <source-l4-port>	Specifies the Layer 4 source port number
action forward-class <class>	class-0, class-1, class-2, class-3, class-4, class-5, class-6, class-7
action remark-codepoint <remark-codepoint>	Specifies the remark-map codepoint value

In the following example, the l3v6 acl-type multifield classifier is configured with qualifier/match based on ipv6-tc (128, 160) and source-ipv6-prefix (132:1:1::3/32) with action as forward-class (class-0,class-1).

```
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 match
ipv6-tc 128
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 match
source-ipv6-prefix 132::1::3/32
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 action
forward-class class-0
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 match
ipv6-tc 160
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 match
source-ipv6-prefix 132:1:1::3/32
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 action
forward-class class-1
commit
```

The following example shows the IPv6 match configuration for the multifield classifier:

```
supervisor@rtbrick>multiservice-edge.rtbrick.net: cfg> show config forwarding-options class-of-service
multifield-classifier acl l3v6 rule global_mfc
{
    "rtbrick-config:rule": [
      {
        "rule-name": "global_mfc",
        "ordinal": [
          {
            "ordinal-value": 1001,
            "match": {
              "ipv6-tc": 128,
              "source-ipv6-prefix": "132:1:1::3/32"
            },
            "action": {
              "forward-class": "class-0"
            }
          },
          {
            "ordinal-value": 1002,
            "match": {
              "ipv6-tc": 160,
              "source-ipv6-prefix": "132:1:1::3/32"
            },
            "action": {
              "forward-class": "class-1"
            }
          }
        ]
      }
    ]
  }
```

IPv4/IPv6 Priority Configuration

Use the following CLI syntax to configure the IPv4/IPv6 priority configuration:

set forwarding-options class-of-service multifield-classifier acl [**l3v4** | **l3v6**] **rule** <rule-name> <attribute> <value>

Attribute	Description
<rule-name>	Specifies the multifield classifier rule name
ordinal <ordinal-value>	Specifies the ordinal that is used for traffic policy rule referencing
ordinal <ordinal-value> <priority>	Specify the ACL priority value. Range: 0 - 65535.

The following example configures the l3v6 acl-type multifield classifier with qualifier/match based on ipv6-tc (128) and source-ipv6-prefix(132:1:1::3/32) with action as forward-class (class-0,class-1) along with priority value (100, 200). When there are multiple qualifiers or actions, the one with the higher priority takes precedence over the others.

```
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 match
ipv6-tc 128
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 match
source-ipv6-prefix 132::1::3/32
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 priority
100
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1001 action
forward-class class-0
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 match
ipv6-tc 128
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 match
source-ipv6-prefix 132:1:1::3/32
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 priority
200
set forwarding-options class-of-service multifield-classifier acl l3v6 rule global_mfc ordinal 1002 action
forward-class class-1
commit
```

The following example shows the IPv4 priority configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service multifield-classifier acl
l3v4 rule rtb_mfc


supervisor@rtbrick>multiservice-edge.rtbrick.net: cfg> show config forwarding-options class-of-service
multifield-classifier acl l3v6 rule global_mfc
{
    "rtbrick-config:rule": [
      {
        "rule-name": "global_mfc",
        "ordinal": [
          {
            "ordinal-value": 1001,
            "priority": 100,
            "match": {
              "ipv6-tc": 128,
              "source-ipv6-prefix": "132:1:1::3/32"
            },
            "action": {
              "forward-class": "class-0"
            }
          },
          {
            "ordinal-value": 1002,
            "priority": 200,
            "match": {
              "ipv6-tc": 160,
              "source-ipv6-prefix": "132:1:1::3/32"
            },
            "action": {
              "forward-class": "class-1"
            }
          }
        ]
      }
    ]
  }
supervisor@rtbrick>LEAF01: cfg>
```

13.8.1. L2 ACL Priority Bit Egress Classifier Configuration

Use the following CLI syntax to configure the L2 ACL priority bit egress classifier.

set forwarding-options <attribute> <value>

Attribute	Description
acl l2 <…>	Specifies the l2 ACL rule for egress priority bit classifier configurations. For more information on configuring the ACL match rules, see the sections below.

Service VLAN Priority Configuration

Use the following CLI syntax to configure the L2 ACL Match Configuration for the egress priority bit classifier:

set forwarding-options acl l2 rule <rule-name> ordinal <ordinal-value> <attribute> <Value>

Attribute	Description
<rule-name>	Specifies the L2 ACL classifier rule name
ordinal <ordinal-value>	Specifies the ordinal that is used for traffic rule referencing
match <direction>	Specifies the direction of L2 traffic
match <service-outer-vlan-priority>	Specifies the outer VLAN priority of the L2 service traffic
action forward-class <class>	class-0, class-1, class-2, class-3, class-4, class-5, class-6, class-7

L2 ACL classification commonly matches on layer-2 fields such as service-inner-vlan-priority, service-outer-vlan-priority, interface-meta-data, and direction

The l2 acl type egress priority bit classifier is configured with a qualifier/match based on service-inner-vlan-priority ` (0-7), interface-meta-data (type-1), direction (ingress), service-outer-vlan-priority (0-7) with action as forward-class (class-0 to class-7).

```
set forwarding-options acl l2 rule evpn_vpws ordinal 0
set forwarding-options acl l2 rule evpn_vpws ordinal 0 match direction ingress
set forwarding-options acl l2 rule evpn_vpws ordinal 0 match service-outer-vlan-priority 0
set forwarding-options acl l2 rule evpn_vpws ordinal 0 match interface-meta-data type-1
set forwarding-options acl l2 rule evpn_vpws ordinal 0 action forward-class class-0
set forwarding-options acl l2 rule evpn_vpws ordinal 1
set forwarding-options acl l2 rule evpn_vpws ordinal 1 match direction ingress
set forwarding-options acl l2 rule evpn_vpws ordinal 1 match service-outer-vlan-priority 1
set forwarding-options acl l2 rule evpn_vpws ordinal 1 match interface-meta-data type-1
set forwarding-options acl l2 rule evpn_vpws ordinal 1 action forward-class class-1
commit
```

The following example shows the L2 ACL P-Bit Egress Classifier configuration:

```
supervisor@rtbrick>multiservice-edge.rtbrick.net: cfg> show config forwarding-options acl l2 rule evpn_vpws
ordinal
{
  "rtbrick-config:ordinal": [
    {
      "ordinal-value": 0,
      "match": {
        "direction": "ingress",
        "service-outer-vlan-priority": 0,
        "interface-meta-data": "type-1"
      },
      "action": {
        "forward-class": "class-0"
      }
    },
    {
      "ordinal-value": 1,
      "match": {
        "direction": "ingress",
        "service-outer-vlan-priority": 1,
        "interface-meta-data": "type-1"
      },
      "action": {
        "forward-class": "class-1"
      }
    }
  ]
}
```

13.8.2. Policer Configuration

Use the following CLI syntax to configure the QoS policer:

set forwarding-options class-of-service policer <policer-name> <attribue> <value>

Attribute	Description
<policer-name>	Specifies the policer name.
<levels>	Specifies levels in the policer. There is only support for the policer levels 1 and 4.

Attribute	Description
<type>	Specifies the policer type.
<flag>	Specifies the policer flags.
level1-rates cir <cir>	Specifies the committed information rate (CIR) in Kilobits per second (Kbps) for level-1. The same is applicable for level-2 to level-4.
level1-rates pir <pir>	Specifies the peak information rate (PIR) in Kilobits per second (Kbps) for level-1. The same is applicable for level-2 to level-4.
level1-rates cbs <cbs>	Specifies the Committed burst size (CBS) in Kilobits per second (Kbps) for level-1. The same is applicable for level-2 to level-4.
level1-rates pbs <pbs>	Specifies the peak burst size (PBS) in Kilobits per second (Kbps) for level-1. The same is applicable for level-2 to level-4.
level1-rates max-cir <max-cir>	Specifies the maximum for the level-1 committed information rate (CIR) in kilobits per second (Kbps). The same is applicable for level-2 to level-4.
level1-rates max-pir <max-pir>	Specifies the maximum for the level-1 peak information rate (PIR) in Kilobits per second (Kbps). The same is applicable for level-2 to level-4.

The following example configures the QoS policer policer-residential with multi-level policers (levels=4), CIR, PIR rates, and burst sizes (CBS, PBS) for each level (level-1, level-2, level-3 and level-4). Also, two-rate-three-color policer type and color-blind as the default configuration. The four-level Policer configuration is as follows:

- Level-1(cir=2Mbps, pir=2.5Mbps, cbs=1000, pbs=1000)
- Level-2(cir=3Mbps, pir=3.5Mbps, cbs=1000, pbs=1000)
- Level-3(cir=4Mbps, pir=4.5Mbps, cbs=1000, pbs=1000)
- Level-4(cir=1Mbps, pir=1.5Mbps, cbs=1000, pbs=1000)

The following configurations show policer configurations for levels=4.

```
set forwarding-options class-of-service policer policer-residential
set forwarding-options class-of-service policer policer-residential level1-rates cir 2000
set forwarding-options class-of-service policer policer-residential level1-rates cbs 1000
set forwarding-options class-of-service policer policer-residential level1-rates pir 2500
set forwarding-options class-of-service policer policer-residential level1-rates pbs 1000
set forwarding-options class-of-service policer policer-residential level2-rates cir 3000
set forwarding-options class-of-service policer policer-residential level2-rates cbs 1000
set forwarding-options class-of-service policer policer-residential level2-rates pir 3500
set forwarding-options class-of-service policer policer-residential level2-rates pbs 1000
set forwarding-options class-of-service policer policer-residential level3-rates cir 4000
set forwarding-options class-of-service policer policer-residential level3-rates cbs 1000
set forwarding-options class-of-service policer policer-residential level3-rates pir 4500
set forwarding-options class-of-service policer policer-residential level3-rates pbs 1000
set forwarding-options class-of-service policer policer-residential level4-rates cir 1000
set forwarding-options class-of-service policer policer-residential level4-rates cbs 1000
set forwarding-options class-of-service policer policer-residential level4-rates pir 1500
set forwarding-options class-of-service policer policer-residential level4-rates pbs 1000
set forwarding-options class-of-service policer policer-residential levels 4
set forwarding-options class-of-service policer policer-residential type two-rate-three-color
commit
```

The following example shows the QoS policer level (levels=4) rates configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service policer policer-residential
{
  "rtbrick-config:policer": [
{
            "policer-name": "policer-residential",
            "level1-rates": {
              "cir": 2000,
              "cbs": 1000,
              "pir": 2500,
              "pbs": 1000
            },
            "level2-rates": {
              "cir": 3000,
              "cbs": 1000,
              "pir": 3500,
              "pbs": 1000
            },
            "level3-rates": {
              "cir": 4000,
              "cbs": 1000,
              "pir": 4500,
              "pbs": 1000
            },
            "level4-rates": {
              "cir": 1000,
              "cbs": 1000,
              "pir": 1500,
              "pbs": 1000
            },
            "levels": 4,
            "type": "two-rate-three-color"
          }
  ]
}
```

The following configurations show policer configurations for levels=1.

```
set forwarding-options class-of-service policer policer-residential
set forwarding-options class-of-service policer policer-residential level1-rates cir 2000
set forwarding-options class-of-service policer policer-residential level1-rates cbs 1000
set forwarding-options class-of-service policer policer-residential level1-rates pir 2500
set forwarding-options class-of-service policer policer-residential level1-rates pbs 1000
set forwarding-options class-of-service policer policer-residential levels 1
set forwarding-options class-of-service policer policer-residential type two-rate-three-color
commit
```

The following example shows the QoS policer level rates (levels=1) configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service policer policer-residential
{
  "rtbrick-config:policer": [
{
            "policer-name": "policer-residential",
            "level1-rates": {
              "cir": 2000,
              "cbs": 1000,
              "pir": 2500,
              "pbs": 1000
            }
            "levels": 1,
            "type": "two-rate-three-color"
          }
  ]
}
```

13.8.3. Class-Policer-Map Configuration

Use the following CLI syntax to configure the Class-Policer-Map:

set forwarding-options class-of-service class-policer-map <class-policer-map-name> <attribue> <value>

Attribute	Description
<class-policer-map-name>	Specifies the class policer map name,
class <class>	Specifies the class such as class-0, class-1, class-2, class-3, class-4, class-5, class-6, class-7
class <class> <policer-level>	Specifies the policer levels. The supported levels are: level-1, level-2, level-3, and level-4

Below is an example configuration of the class-policer-map, which sets up level-1 to level-4 policer levels.

- class-0 mapped to policer level-1
- class-1 mapped to policer level-2
- class-2 mapped to policer level-3
- class-3 mapped to policer level-4

```
set forwarding-options class-of-service class-policer-map policer-map-residential class class-0 policer-level
level-1
set forwarding-options class-of-service class-policer-map policer-map-residential class class-1 policer-level
level-2
set forwarding-options class-of-service class-policer-map policer-map-residential class class-2 policer-level
level-3
set forwarding-options class-of-service class-policer-map policer-map-residential class class-3 policer-level
level-4
commit
```

The following example shows the class-policer-map configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service class-policer-map policer-
map-l2tp
{
  "rtbrick-config:class-policer-map": [
{
            "class-policer-map-name": "policer-map-residential",
            "class": [
              {
                "class": "class-0",
                "policer-level": "level-1"
              },
              {
                "class": "class-1",
                "policer-level": "level-2"
              },
              {
                "class": "class-2",
                "policer-level": "level-3"
              },
              {
                "class": "class-3",
                "policer-level": "level-4"
              }
            ]
          }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

13.8.4. Queue Configuration

Use the following CLI syntax to configure a queue:

set forwarding-options class-of-service queue <queue-name> <atribute> <value>

Attribute	Description
<queue-name>	Specifies the queue name
queue-size <queue-size>	Specifies the size of the queue in bytes
<shaper-name>	(Optional) Specifies the shaper that is associated with the queue

Attribute	Description
wred minimum-threshold <minimum-threshold>	Specifies the minimum average queue size to apply WRED in bytes
wred maximum-threshold <maximum-threshold>	Specifies the maximum average queue size to apply WRED in bytes
wred drop-probability <drop-probability>	WRED drop probability applied at the maximum threshold
header-compensation bytes <bytes>	Specifies the header compensation value
header-compensation decrement true	Specifies whether the header compensation value is to be decremented.

The following example configures four queues with different traffic streams named BE_SUBS, LD_SUBS, LL_SUBS and VO_SUBS with queue attributes queue-size, queue header-compensation & queue shaper for Best Effort(BE), Low-Delay(LD), Low-Loss(LL) and Voice(VO) Queues.

```
set forwarding-options class-of-service queue BE_SUBS
set forwarding-options class-of-service queue BE_SUBS queue-size 375000
set forwarding-options class-of-service queue BE_SUBS header-compensation bytes 22
set forwarding-options class-of-service queue BE_SUBS header-compensation decrement true
set forwarding-options class-of-service queue LD_SUBS
set forwarding-options class-of-service queue LD_SUBS queue-size 625000
set forwarding-options class-of-service queue LD_SUBS header-compensation bytes 22
set forwarding-options class-of-service queue LD_SUBS header-compensation decrement true
set forwarding-options class-of-service queue LL_SUBS
set forwarding-options class-of-service queue LL_SUBS queue-size 625000
set forwarding-options class-of-service queue LL_SUBS header-compensation bytes 22
set forwarding-options class-of-service queue LL_SUBS header-compensation decrement true
set forwarding-options class-of-service queue VO_SUBS
set forwarding-options class-of-service queue VO_SUBS queue-size 156250
set forwarding-options class-of-service queue VO_SUBS header-compensation bytes 22
set forwarding-options class-of-service queue VO_SUBS header-compensation decrement true
set forwarding-options class-of-service queue VO_SUBS shaper-name shaper_VO
commit
```

The following example shows the queue configuration:

```
supervisor@DT-STD-23-2402>bm14-tst.fsn.rtbrick.net: cfg> show config forwarding-options class-of-service
queue
{
  "rtbrick-config:queue": [
{
            "queue-name": "BE_SUBS",
            "queue-size": 375000,
            "header-compensation": {
              "bytes": 22,
              "decrement": "true"
            }
```

```
          },
          {
            "queue-name": "LD_SUBS",
            "queue-size": 625000,
            "header-compensation": {
              "bytes": 22,
              "decrement": "true"
            }
          },
          {
            "queue-name": "LL_SUBS",
            "queue-size": 625000,
            "header-compensation": {
              "bytes": 22,
              "decrement": "true"
            }
          },
          {
            "queue-name": "VO_SUBS",
            "queue-size": 156250,
            "shaper-name": "shaper_VO",
            "header-compensation": {
              "bytes": 22,
              "decrement": "true"
            }
          }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

13.8.5. Queue-Group Configuration

Queue group size: 1 or 4 or 8

Use the following CLI syntax to configure a queue group:

set forwarding-options class-of-service queue-group <queue-group-name> <attribute> <value>

Attribute	Description
<queue-group-name>	User-defined name for the queue-group
queue-numbers <queue-numbers>	Specifies the number of queues in a Queue Group

The following examples configure the queue group with queue numbers 1 and 4.

```
set forwarding-options class-of-service queue-group subs-4queues queue-numbers 1
commit
```

The following example shows the queue group configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service queue-group
{
  "rtbrick-config:queue-group": [
    {
      "queue-group-name": "subs-4queues",
      "queue-numbers": 1
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

```
set forwarding-options class-of-service queue-group subs-4queues queue-numbers 4
```

The following example shows the queue group configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service queue-group
{
  "rtbrick-config:queue-group": [
    {
      "queue-group-name": "subs-4queues",
      "queue-numbers": 4
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

Class-Queue-Map Configuration

Use the following CLI syntax to configure the class-queue-map:

set forwarding-options class-of-service class-queue-map <class-queue-map-name> <attribute> <value>

Attribute	Description
<class-queue-map-name>	Specifies the class queue map name
class <class>	Specifies the class such as class-0, class-1, class-2, class-3, class-4, class-5, class-6, class-7
class <class> queue-name <queue-name>	Specifies the queue name

The following example configures the class-queue-map for the specific classes and queues:

- class-0 mapped to queue BE_SUBS

- class-1 mapped to queue LD_SUBS
- class-2 mapped to queue LL_SUBS
- class-3 mapped to queue VO_SUBS

```
set forwarding-options class-of-service class-queue-map subs-4queues class class-0 queue-name BE_SUBS
set forwarding-options class-of-service class-queue-map subs-4queues class class-1 queue-name LD_SUBS
set forwarding-options class-of-service class-queue-map subs-4queues class class-2 queue-name LL_SUBS
set forwarding-options class-of-service class-queue-map subs-4queues class class-3 queue-name VO_SUBS
commit
```

The following example shows the class-queue-map configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service class-queue-map
{
  "rtbrick-config:class-queue-map": [
{
            "class-queue-map-name": "subs-4queues",
            "class": [
              {
                "class-type": "class-0",
                "queue-name": "BE_SUBS"
              },
              {
                "class-type": "class-1",
                "queue-name": "LD_SUBS"
              },
              {
                "class-type": "class-2",
                "queue-name": "LL_SUBS"
              },
              {
                "class-type": "class-3",
                "queue-name": "VO_SUBS"
              }
            ]
          }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

13.8.6. Scheduler Configuration

Use the following CLI syntax to configure a scheduler:

set forwarding-options class-of-service scheduler <attribue> <value>

Attribute	Description
<scheduler-name>	User-defined Scheduler Name

Attribute	Description
<scheduler-name> shaper-name <shaper-name>	(Optional) User-defined Shaper Name
<scheduler-name> type <type>	Specifies the Scheduler Type 2sp_wfq_discrete 3sp_wfq_discrete strict_priority wfq_discrete_2sp 2sp_wfq_independent fair_queueing weighted_fair_queueing wfq_independent_2sp
<scheduler-name> composite true	(Optional) keyword to specify the scheduler as composite type

The following example configures two schedulers: subs-4queues with scheduler type strict_priority for the Subscriber level, and pon0 with scheduler type fair_queueing for PON/GPON level.

```
set forwarding-options class-of-service scheduler subs-4queues
set forwarding-options class-of-service scheduler subs-4queues shaper shaper_session
set forwarding-options class-of-service scheduler subs-4queues type strict_priority
set forwarding-options class-of-service scheduler pon0
set forwarding-options class-of-service scheduler pon0 type fair_queueing
commit
```

The following example shows the QoS scheduler configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service scheduler
{
  "rtbrick-config:scheduler": [
          {
            "scheduler-name": "pon0",
            "type": "fair_queueing"
          },
          {
            "scheduler-name": "subs-4queues",
            "shaper-name": "shaper_session",
            "type": "strict_priority"
          }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

Priority Propagation

Use the following CLI syntax to configure priority propagation:

set forwarding-options class-of-service global priority-propagation [**enable** | **disable**]

The following example enables priority propagation at the global level to operate schedulers in dual-flow mode, with high-priority and low-priority flows.

```
set forwarding-options class-of-service global priority-propagation enable
```

The following example shows the priority propagation:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service global
{
  "rtbrick-config:global": {
    "priority-propagation": "enable"
  }
}
```

13.8.7. Shaper Configuration

Use the following CLI syntax to configure a shaper:

set forwarding-options class-of-service shaper <shaper-name> <attribue> <value>

Command Arguments

Attribute	Description
<shaper-name>	User-defined shaper name
<shaper-name> shaping-rate-high <0-700000000>	High flow shaping rate in kilobits per second

Attribute	Description
<shaper-name> shaping-rate-high-precision true	Configures the precision of high-flow shapers, which can only be used with session shapers.
<shaper-name> shaping-rate-low <0-700000000>	Low flow shaping rate in kilobits per second

- Low-rate shaping is only supported for high-priority flows. Enabling low-rate shaping consumes additional QoS resources, which will impact the scalability.
- If priority propagation is not enabled, high-flow shaping value will be considered for shaper.
- If the scheduler type is strict_priority, the mapping of queues to priorities begins with strict_priority_1.

The following example configures the shaper high-flow and low-flow rates for Subscriber Session Level with shaper-name shaper_session and Queue Level with shaper-name shaper_VO.

```
set forwarding-options class-of-service shaper shaper_session
set forwarding-options class-of-service shaper shaper_session shaping-rate-high 10000
set forwarding-options class-of-service shaper shaper_session shaping-rate-low 100
set forwarding-options class-of-service shaper shaper_VO
set forwarding-options class-of-service shaper shaper_VO shaping-rate-high 2000
set forwarding-options class-of-service shaper shaper_VO shaping-rate-low 0
commit
```

The following example shows the shaper configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service shaper
{
  "rtbrick-config:shaper": [
          {
            "shaper-name": "shaper_VO",
            "shaping-rate-high": 2000,
            "shaping-rate-low": 0
          },
          {
            "shaper-name": "shaper_session",
            "shaping-rate-high": 10000,
            "shaping-rate-low": 100
          }
  ]
}
```

```
supervisor@rtbrick>LEAF01: cfg>
```

With the following shaper configuration, the low flow is limited to 1Mbps and the high flow is limited to 512Kbps. This results in a session rate of approximately 1.5Mbps when 10Mbps is distributed between high and low flows.

```
set forwarding-options class-of-service shaper session-shaper
set forwarding-options class-of-service shaper session-shaper shaping-rate-high-
precision true
set forwarding-options class-of-service shaper session-shaper shaping-rate-high
512
set forwarding-options class-of-service shaper session-shaper shaping-rate-low
1000
```

The following example shows the shaper configuration, where the high precision shaping is enabled:

```
{
  "rtbrick-config:shaper": [
    {
      "shaper-name": "session-shaper",
      "shaping-rate-high": 512,
      "shaping-rate-high-precision": "true",
      "shaping-rate-low": 1000
   }
  ]
}
```

13.8.8. Scheduler-Map Configuration

Use the following CLI syntax to configure a Scheduler-Map:

set forwarding-options class-of-service scheduler-map <scheduler-map-name> <attribue> <value>

Command arguments

Attribute	Description
<scheduler-map-name>	Specifies the name of the scheduler-map
scheduler-name <scheduler-name>	Specifies the name of the scheduler

Attribute	Description
queue-group-name <group-name>	Specifies the name of the queue-group
queue-group-name <group-name> queue-name <name>	Specifies the name of the queue
queue-group-name <group-name> queue-name <name> connection-point <connection-point>	Specifies the type of connection point, such as no_priority, strict_priority_0, strict_priority_1, strict_priority_2, strict_priority
queue-group-name <group-name> queue-name <name> parent-flow <high-flow / low-flow>	(Optional) Specifies the type of the parent flow, that is high-flow or low-flow.
queue-group-name <group-name> queue-name <name> parent-scheduler-name <parent-scheduler-name>	Specifies the name of the parent scheduler
queue-group-name <group-name> queue-name <name> port-connection	<port-connection-type>
Specifies the type of port connection, that is, queue_to_port or scheduler_to_port	queue-group-name <group-name> queue-name <name> weight <weight>

The following example configures a scheduler map for OLT with schedmap-olt and for subscribers with subs-4queues-residential. OLT scheduler-map schedmap-olt is directly connected to the physical port and subs-4queues-residential connects different queues (BE_SUBS, LL_SUBS,LD_SUBS,VO_SUBS) with different connection-points(strict_priority_3, strict_priority_2, strict_priority_1, strict_priority_0).

```
set forwarding-options class-of-service scheduler-map schedmap-olt
set forwarding-options class-of-service scheduler-map schedmap-olt scheduler-name pon0
set forwarding-options class-of-service scheduler-map schedmap-olt scheduler-name pon0 port-connection
scheduler_to_port
set forwarding-options class-of-service scheduler-map subs-4queues-residential
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name BE_SUBS
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name BE_SUBS parent-flow high-flow
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name BE_SUBS parent-scheduler-name subs-4queues
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name BE_SUBS connection-point strict_priority_3
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LD_SUBS
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LD_SUBS parent-flow high-flow
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LD_SUBS parent-scheduler-name subs-4queues
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LD_SUBS connection-point strict_priority_1
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LL_SUBS
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LL_SUBS parent-flow high-flow
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LL_SUBS parent-scheduler-name subs-4queues
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name LL_SUBS connection-point strict_priority_2
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name VO_SUBS
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name VO_SUBS parent-flow high-flow
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name VO_SUBS parent-scheduler-name subs-4queues
set forwarding-options class-of-service scheduler-map subs-4queues-residential queue-group-name subs-4queues
queue-name VO_SUBS connection-point strict_priority_0
set forwarding-options class-of-service scheduler-map subs-4queues-residential scheduler-name subs-4queues
set forwarding-options class-of-service scheduler-map subs-4queues-residential scheduler-name subs-4queues
port-connection scheduler_to_port
commit
```

The following example shows the scheduler-map configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service scheduler-map
{
  "rtbrick-config:scheduler-map": [
{
            "scheduler-map-name": "schedmap-olt",
            "scheduler-name": [
              {
                "name": "pon0",
                "port-connection": "scheduler_to_port"
              }
            ]
          },
          {
            "scheduler-map-name": "subs-4queues-residential",
            "queue-group-name": [
              {
                "group-name": "subs-4queues",
                "queue-name": [
                  {
                    "name": "BE_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_3"
                  },
                  {
                    "name": "LD_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_1"
                  },
                  {
                    "name": "LL_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_2"
                  },
                  {
                    "name": "VO_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_0"
                  }
                ]
              }
            ],
            "scheduler-name": [
              {
                "name": "subs-4queues",
                "port-connection": "scheduler_to_port"
              }
            ]
          }
    ]
  }
```

Remark-Map Configuration

Use the following CLI syntax to configure the remark-map:

set forwarding-options class-of-service remark-map <remark-map-name> <attribute> <value>

Command arguments

Attribute	Description
<remark-map-name>	Specifies the remaking map name.

Attribute	Description
remark-type <remark-type>	Specifies the remarking type - ipv4-tos, ipv6-tc, mpls-ipv4, mpls-ipv6, ieee-802.1, class-to-ip, ip-tos. The "ip-tos" remark-type can be used to set the IPv4 ToS or IPv6 TC values in IPv4 or IPv6 packet header respectively.
remark-type <remark-type> <match-codepoint>	Specifies the match code point for the specified remarking type. On the UfiSpace S9600-72XC and UfiSpace S9600-32X platforms, the match codepoint is TOS for VLAN IEEE-802.1p remarking.
remark-type <remark-type> <match-codepoint> color <color>	Indicates the color - all, green, yellow. Color is used to set different remark codepoints for the same match-codepoint based on color marked by the Policer.
remark-type <remark-type> <match-codepoint> color <color> remark-codepoint <remark-codepoint>	Specifies the remarking codepoint

In the following example, the remark map subs-remarking-residential is configured with match-codepoint as 128, 160, 192, and 224. The color is set to "all", and the remark-codepoint is 6.

```
set forwarding-options class-of-service remark-map subs-remarking-residential
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 128
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 128 color all
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 128 color all remark-codepoint 6
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 160
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 160 color all
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 160 color all remark-codepoint 6
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 192
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 192 color all
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 192 color all remark-codepoint 6
```

```
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 224
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 224 color all
set forwarding-options class-of-service remark-map subs-remarking-residential remark-type ieee-802.1 match-
codepoint 224 color all remark-codepoint 6
commit
```

The following example shows the remark map configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service remark-map remark-exp remark-
type ipv6-tc
{
  "rtbrick-config:remark-type": [
{
            "remark-map-name": "subs-remarking-residential",
            "remark-type": [
              {
                "remark-type": "ieee-802.1",
                "match-codepoint": [
                  {
                    "match-codepoint": 128,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  },
                  {
                    "match-codepoint": 160,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  },
                  {
                    "match-codepoint": 192,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  },
                  {
                    "match-codepoint": 224,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  }
                ]
              }
            ]
          }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

Global Profile Mapping

Use the following CLI syntax to configure the remark map for a global profile.

set forwarding-options class-of-service global remark-map-name <remark-map-name>

Command arguments

Attribute	Description
<remark-map-name>	Specifies the name of the remark map.

The following example configures the remark map for a global profile:

```
set forwarding-options class-of-service global remark-map-name subs-remarking-residential
```

The following example shows the remark map for a global profile configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service global
{
  "rtbrick-config:global": {
    "multifield-classifier-name": "global_mfc",
    "remark-map-name": "subs-remarking-residential"
  }
}
supervisor@rtbrick>LEAF01: cfg>
```

Remark-map to Instance Mapping

Use the following CLI syntax to configure the remark map for an instance.

set forwarding-options class-of-service instance <instance-name> **remark-map-name** <remark-map-name>

Command arguments

Attribute	Description
<instance-name>	Specifies the name of the instance.
<remark-map-name>	Specifies the name of the remark map.

The following example configures the remark map for the instance default.

```
set forwarding-options class-of-service instance default remark-map-name subs-remarking-residential
commit
```

The following example shows the remark map configuration for the instance default.

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service instance default
{
  "rtbrick-config:instance": [
    {
      "name": "default",
      "remark-map-name": "subs-remarking-residential"
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

Configuring QoS L2 and L3 Logical Interfaces

QoS Configuration

To support QoS on L2 logical interface, the class-of-service profile configuration is introduced. If a L2 logical interface is part of an interface-set, logical-interface scheduler/shaper must have parent-scheduler as interface-set-scheduler/shaper.

```
set interface ifp-0/0/16 unit 201
set interface ifp-0/0/16 unit 201 interface-type l2vpn-vpws
set interface ifp-0/0/16 unit 201 instance evpn-vpws-vrf2
set interface ifp-0/0/16 unit 201 vlan 200
set interface ifp-0/0/16 unit 201 inner-vlan 201
set interface ifp-0/0/16 unit 201 class-of-service qos_profile_service_1
set interface ifp-0/0/16 unit 202
set interface ifp-0/0/16 unit 202 interface-type l2vpn-vpws
set interface ifp-0/0/16 unit 202 instance evpn-vpws-vrf2
set interface ifp-0/0/16 unit 202 vlan 200
set interface ifp-0/0/16 unit 202 inner-vlan 202
set interface ifp-0/0/16 unit 202 class-of-service qos_profile_service_2
set interface ifp-0/0/16 unit 203
set interface ifp-0/0/16 unit 203 interface-type l2vpn-vpws
set interface ifp-0/0/16 unit 203 instance evpn-vpws-vrf2
set interface ifp-0/0/16 unit 203 vlan 200
set interface ifp-0/0/16 unit 203 inner-vlan 203
set interface ifp-0/0/16 unit 203 class-of-service qos_profile_service_3
```

Interface-set Configuration

```
set interface-set service-group-1 interface ifp-0/0/16
set interface-set service-group-1 interface ifp-0/0/16 unit 201
set interface-set service-group-1 interface ifp-0/0/16 unit 202
set interface-set service-group-1 interface ifp-0/0/16 unit 203
```

Combining Multiple Logical Interfaces with an Added QoS Hierarchy

Configuring QoS Profile for a Logical Interface or Service

```
set forwarding-options class-of-service profile qos_profile_service_1
set forwarding-options class-of-service profile qos_profile_service_1 classifier-name upstream_pbit_classifer
set forwarding-options class-of-service profile qos_profile_service_1 class-queue-map-name
business_service_class_queue_map
set forwarding-options class-of-service profile qos_profile_service_1 class-policer-map-name
business_service_class_policer_map
set forwarding-options class-of-service profile qos_profile_service_1 policer-name business_service_policer_1
set forwarding-options class-of-service profile qos_profile_service_1 scheduler-map-name
business_service_scheduler_map_1
```

Configuring QoS Profile for a Logical Interface Set

```
set forwarding-options class-of-service profile qos_profile_service_group
set forwarding-options class-of-service profile qos_profile_service_group scheduler-map-name
business_service_group_scheduler_map
```

Enabling QoS Profile for a Logical Interface Set

```
set interface-set service-group-1
set interface-set service-group-1 class-of-service qos_profile_service_group
set forwarding-options class-of-service profile service-group-profile
set forwarding-options class-of-service profile service-group-profile class-queue-map-name subs-4queues
set forwarding-options class-of-service profile service-group-profile class-policer-map-name policer-map-
residential
set forwarding-options class-of-service profile service-group-profile policer-name policer-residential
set forwarding-options class-of-service profile service-group-profile scheduler-map-name
business_service_group_scheduler_map
```

The following CLIs are used to configure scheduler port connection to link schedulers across different scheduler maps:

QoS Scheduler-map to connect logical interface scheduler with interface-set scheduler:

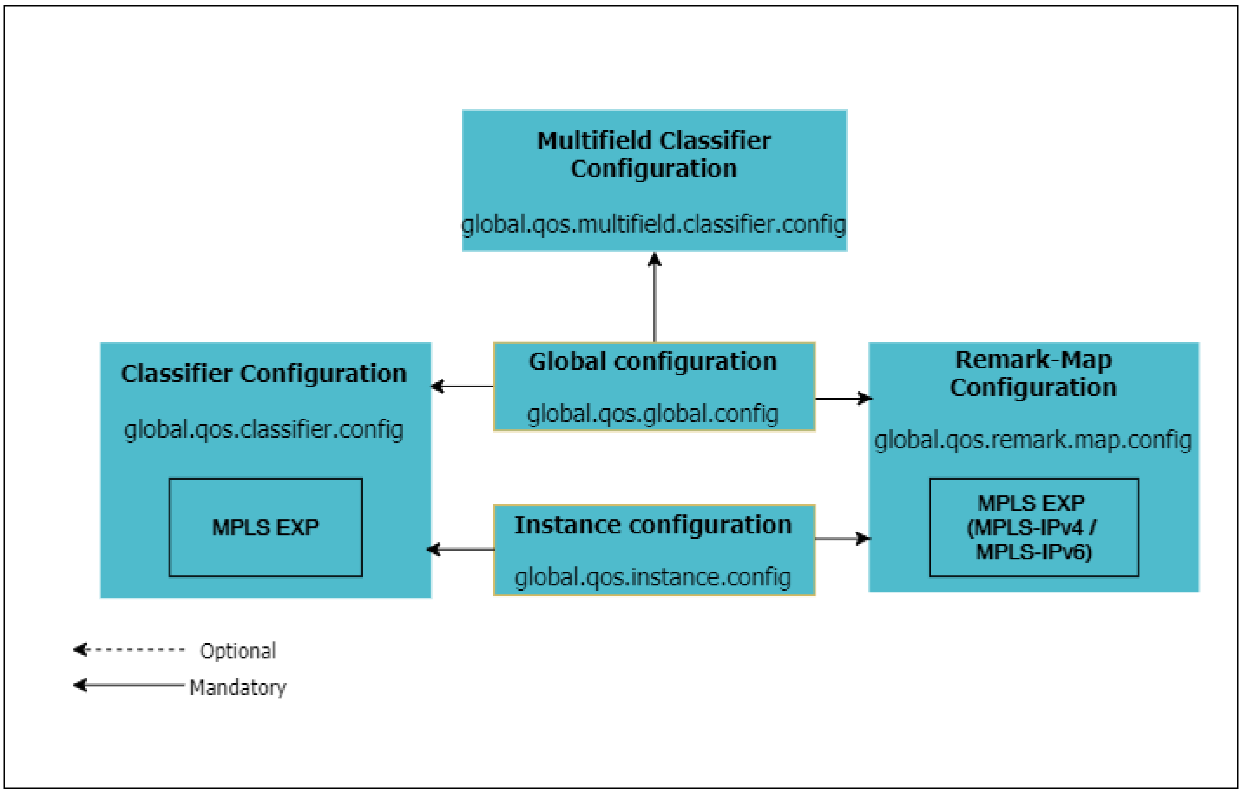
```
set forwarding-options class-of-service scheduler-map schedmap_service_1
set forwarding-options class-of-service scheduler-map schedmap_service_1  scheduler-name sch_service_1
set forwarding-options class-of-service scheduler-map schedmap_service_1  scheduler-name sch_service_1 port-
connection scheduler-map-to-scheduler-map
set forwarding-options class-of-service scheduler-map schedmap_service_1   scheduler-name
business_service_scheduler_1 parent-scheduler-name business_service_group_scheduler
```

QoS Scheduler-map to Connect an interface-set scheduler with a pon scheduler

```
set forwarding-options class-of-service scheduler-map business_service_group_scheduler_map
set forwarding-options class-of-service scheduler-map business_service_group_scheduler_map scheduler-name
business_service_group_scheduler
set forwarding-options class-of-service scheduler-map business_service_group_scheduler scheduler-name
business_service_group_scheduler parent-scheduler-name pon0
set forwarding-options class-of-service scheduler-map business_service_group_scheduler_map scheduler-name
business_service_group_scheduler port-connection scheduler_map_to_scheduler_map
```

QoS Global and Instance Configurations

The figure below shows the dependencies for per instance or global classifier and remark-map configurations.


Multifield Classifier Configuration
global.qos.multifield.classifier.config
Classifier Configuration
global.qos.classifier.config
MPLS EXP
Global configuration
global.qos.global.config
Remark-Map Configuration
global.qos.remark.map.config
Instance configuration
global.qos.instance.config
MPLS EXP (MPLS-IPv4 / MPLS-IPv6)
Optional
Mandatory

The following example configures the BA classifier subs-exp-class for traffic exp with match code point 2 classified as class class-1.

```
set forwarding-options class-of-service classifier subs-exp-class match-type exp codepoint 2 class class-1
```

BA Classifier to Global Mapping

The MPLS classifiers can be applied globally using global configuration.

```
set forwarding-options class-of-service global classifier-name <classifier-name>
```

Attribute	Description
<classifier-name>	Specifies the classifier name

The following example shows how to configure the subs-exp-class BA Classifier for global mapping.

```
set forwarding-options class-of-service global classifier-name subs-exp-class
```

The configuration for global mapping to the BA Classifier is shown in the following example.

```
supervisor@rtbrick>LEAF01: cfg> show config  forwarding-options class-of-service global classifier-name
{
  "rtbrick-config:classifier-name": "subs-exp-class"
}
supervisor@rtbrick>LEAF01: cfg>
```

Single Queue To Global Mapping

The Single-Queue can be enabled with global configuration.

For the system to initialize with a single queue in a Queue-Group, a system reboot is required.

```
set forwarding-options class-of-service global queue-group-profile single-queue
```

Attribute	Description

The following example shows how to enable single-queue in a Queue-Group.

```
set forwarding-options class-of-service global queue-group-profile single-queue
commit
```

The configuration for a single queue in Queue-Group is shown in the following example.

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service global
{
  "rtbrick-config:global": {
    "queue-group-profile": "single-queue"
  }
}
supervisor@rtbrick>LEAF01: cfg>
```

Remark-Map To Global Mapping

Use the following CLI syntax to configure the remark map for a global profile.

set forwarding-options class-of-service global remark-map-name <remark-map-name>

Command arguments

Attribute	Description
<remark-map-name>	Specifies the name of the remark map.

The following example configures the remark map for a global profile:

```
set forwarding-options class-of-service global remark-map-name subs-remarking-residential
commit
```

Multifield Classifier (MFC) to Global Mapping

Use the following CLI syntax to configure the MF Classifier to Global Mapping:

set forwarding-options class-of-service global multifield-classifier-name <multifield-classifier-name>

Attribute	Description
<multifield-classifier-name>	Specifies the name of the multifield classifier

The following example configures the multifield Classifier to Global Mapping:

```
set forwarding-options class-of-service global multifield-classifier-name global_mfc
commit
```

The following example shows the multifield classifier to Global Mapping configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service global
{
  "rtbrick-config:global": {
    "multifield-classifier-name": "global_mfc"
  }
}
supervisor@rtbrick>LEAF01: cfg>
```

BA Classifier to Instance Mapping

The MPLS classifiers can be applied at an instance level using instance configuration.

```
set forwarding-options class-of-service instance <instance> classifier-name <classifier-name>
```

Attribute	Description
<instance>	Specifies the instance name
<classifier-name>	Specifies the classifier name

The following example shows how to configure the subs-exp-class BA Classifier for the instance default.

```
set forwarding-options class-of-service instance default classifier-name subs-exp-class
commit
```

The configuration for instance mapping to the BA Classifier is shown in the following example.

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service instance default
{
  "rtbrick-config:instance": [
    {
      "name": "default",
      "classifier-name": "subs-exp-class"
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

Remark-Map To Instance Mapping

Use the following CLI syntax to configure the remark map for an instance.

set forwarding-options class-of-service instance <instance-name> remark-map-name <remark-map-name>

Command arguments

Attribute	Description
<instance-name>	Specifies the name of the instance.
<remark-map-name>	Specifies the name of the remark map.

The following example configures the remark map for the instance default:

```
set forwarding-options class-of-service instance default remark-map-name subs-remarking-residential
commit
```

The configuration for the remark map for an instance is shown in the following example.

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service instance default
{
  "rtbrick-config:instance": [
    {
      "name": "default",
      "remark-map-name": "subs-remarking-residential"
    }
  ]
}
supervisor@rtbrick>LEAF01: cfg>
```

13.8.9. HQoS Show Running-Configuration

To display the running configuration, use the **show config** command.

Syntax

show config

Example

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options
    "rtbrick-config:forwarding-options": {
      "mirror": [
        {
          "name": "m1",
          "destination": {
            "interface": "cpu-0/0/200"
          },
          "source": {
            "direction": "ingress",
            "interface": "ifp-0/1/30"
          }
        }
      ],
      "class-of-service": {
        "classifier": [
          {
            "classifier-name": "subs-pbit-class",
            "match-type": [
              {
                "match-type": "ieee-802.1",
                "codepoint": [
                  {
                    "codepoint": 1,
                    "class": "class-0",
                    "remark-codepoint": 7
                  },
                  {
                    "codepoint": 2,
                    "class": "class-1",
                    "remark-codepoint": 7
                  },
                  {
                    "codepoint": 3,
                    "class": "class-2",
                    "remark-codepoint": 7
                  },
                  {
                    "codepoint": 4,
                    "class": "class-3",
                    "remark-codepoint": 7
                  }
                ]
              }
            ]
          }
        ],
        "class-policer-map": [
          {
            "class-policer-map-name": "policer-map-residential",
            "class": [
              {
                "class": "class-0",
                "policer-level": "level-1"
              },
              {
                "class": "class-1",
                "policer-level": "level-2"
              },
              {
                "class": "class-2",
                "policer-level": "level-3"
              },
              {
                "class": "class-3",
                "policer-level": "level-4"
              }
            ]
          }
        ],
        "class-queue-map": [
          {
            "class-queue-map-name": "subs-4queues",
            "class": [
              {
                "class-type": "class-0",
                "queue-name": "BE_SUBS"
              },
              {
                "class-type": "class-1",
                "queue-name": "LD_SUBS"
              },
              {
                "class-type": "class-2",
                "queue-name": "LL_SUBS"
              },
              {
                "class-type": "class-3",
                "queue-name": "VO_SUBS"
              }
            ]
          }
        ],
        "global": {
          "multifield-classifier-name": "global_mfc"
        },
        "policer": [
          {
            "policer-name": "policer-residential",
            "level1-rates": {
              "cir": 2000,
              "cbs": 1000,
              "pir": 2500,
              "pbs": 1000
            },
            "level2-rates": {
              "cir": 3000,
              "cbs": 1000,
              "pir": 3500,
              "pbs": 1000
            },
            "level3-rates": {
              "cir": 4000,
              "cbs": 1000,
              "pir": 4500,
              "pbs": 1000
            },
            "level4-rates": {
              "cir": 1000,
              "cbs": 1000,
              "pir": 1500,
              "pbs": 1000
            },
            "levels": 4,
            "type": "two-rate-three-color"
          }
        ],
        "profile": [
          {
            "profile-name": "residential",
            "classifier-name": "subs-pbit-class",
            "class-queue-map-name": "subs-4queues",
            "remark-map-name": "subs-remarking-residential",
            "class-policer-map-name": "policer-map-residential",
            "policer-name": "policer-residential",
            "scheduler-map-name": "subs-4queues-residential"
          }
        ],
        "queue": [
          {
            "queue-name": "BE_SUBS",
            "queue-size": 375000
          },
          {
            "queue-name": "LD_SUBS",
            "queue-size": 625000
          },
          {
            "queue-name": "LL_SUBS",
            "queue-size": 625000
          },
          {
            "queue-name": "VO_SUBS",
            "queue-size": 156250,
            "shaper-name": "shaper_VO"
          }
        ],
        "queue-group": [
          {
            "queue-group-name": "subs-4queues",
            "queue-numbers": 4
          }
        ],
        "remark-map": [
          {
            "remark-map-name": "subs-remarking-residential",
            "remark-type": [
              {
                "remark-type": "ieee-802.1",
                "match-codepoint": [
                  {
                    "match-codepoint": 128,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  },
                  {
                    "match-codepoint": 160,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  },
                  {
                    "match-codepoint": 192,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  },
                  {
                    "match-codepoint": 224,
                    "color": [
                      {
                        "color": "all",
                        "remark-codepoint": 6
                      }
                    ]
                  }
                ]
              }
            ]
          }
        ],
        "scheduler": [
          {
            "scheduler-name": "pon0",
            "shaper-name": "gpon-shaper",
            "type": "fair_queueing"
          },
          {
            "scheduler-name": "subs-4queues",
            "shaper-name": "shaper_session",
            "type": "strict_priority",
            "composite": "false"
          }
        ],
        "scheduler-map": [
          {
            "scheduler-map-name": "schedmap-olt",
            "scheduler-name": [
              {
                "name": "pon0",
                "port-connection": "scheduler_to_port"
              }
            ]
          },
          {
            "scheduler-map-name": "subs-4queues-residential",
            "queue-group-name": [
              {
                "group-name": "subs-4queues",
                "queue-name": [
                  {
                    "name": "BE_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_3"
                  },
                  {
                    "name": "LD_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_1"
                  },
                  {
                    "name": "LL_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_2"
                  },
                  {
                    "name": "VO_SUBS",
                    "parent-flow": "high-flow",
                    "parent-scheduler-name": "subs-4queues",
                    "connection-point": "strict_priority_0"
                  }
                ]
              }
            ],
            "scheduler-name": [
              {
                "name": "subs-4queues",
                "port-connection": "scheduler_to_port"
              }
            ]
          }
        ],
        "shaper": [
          {
            "shaper-name": "shaper_VO",
            "shaping-rate-high": 2000,
            "shaping-rate-low": 0
          },
          {
            "shaper-name": "shaper_session",
            "shaping-rate-high": 10000,
            "shaping-rate-low": 100
          },
          {
            "shaper-name": "gpon-shaper",
            "shaping-rate-high": 2488000,
            "shaping-rate-low": 32000
          }
        ],
        "multifield-classifier": {
          "acl": {
            "l3v4": {
              "rule": [
                {
                  "rule-name": "global_mfc",
                  "ordinal": [
                    {
                      "ordinal-value": 1001,
                      "match": {
                        "ipv4-tos": 128,
                        "source-ipv4-prefix": "192.0.2.2/32"
                      },
                      "action": {
                        "forward-class": "class-0"
                      }
                    },
                    {
                      "ordinal-value": 1002,
                      "match": {
                        "ipv4-tos": 160,
                        "source-ipv4-prefix": "192.0.2.2/32"
                      },
                      "action": {
                        "forward-class": "class-1"
                      }
                    },
                    {
                      "ordinal-value": 1003,
                      "match": {
                        "ipv4-tos": 192,
                        "source-ipv4-prefix": "192.0.2.2/32"
                      },
                      "action": {
                        "forward-class": "class-2"
                      }
                    },
                    {
                      "ordinal-value": 1004,
                      "match": {
                        "ipv4-tos": 224,
                        "source-ipv4-prefix": "192.0.2.2/32"
                      },
                      "action": {
                        "forward-class": "class-3"
                      }
                    }
                  ]
                }
              ]
            }
          }
        }
      }
    }
```

13.8.10. HQoS Operational Commands

HQoS Show Commands

The HQoS show commands provide detailed information about the HQoS operation.

show qos classifier

Syntax:

show qos classifier <option>

Option	Description
-	Without any option, this command displays a summary of all the classifiers.
<classifier-name>	Displays QoS classifier information for the specified classifier.

The following example displays a summary of all the HQoS classifiers.

```
supervisor@rtbrick>LEAF01: op> show qos classifier
Classifier: residential-ip-classifier
Active: False
  Match Type   Codepoint    Class      Remark Codepoint    Color
  ipv4-tos     0            class-0    -                   -
  ipv4-tos     32           class-1    -                   -
  ipv4-tos     64           class-2    -                   -
  ipv4-tos     96           class-3    -                   -
  ipv4-tos     128          class-4    -                   -
  ipv4-tos     160          class-5    -                   -
  ipv4-tos     192          class-6    -                   -
  ipv4-tos     224          class-7    -                   -
  ipv6-tc      0            class-0    -                   -
  ipv6-tc      32           class-1    -                   -
  ipv6-tc      64           class-2    -                   -
  ipv6-tc      96           class-3    -                   -
  ipv6-tc      128          class-4    -                   -
  ipv6-tc      160          class-5    -                   -
  ipv6-tc      192          class-6    -                   -
  ipv6-tc      224          class-7    -                   -
Classifier: residential-pbit-classifier
Active: True
  Match Type   Codepoint    Class      Remark Codepoint    Color
  ieee-802.1   0            class-0    -                   -
  ieee-802.1   1            class-1    -                   -
  ieee-802.1   2            class-2    -                   -
  ieee-802.1   3            class-3    -                   -
  ieee-802.1   4            class-4    -                   -
  ieee-802.1   5            class-5    -                   -
  ieee-802.1   6            class-6    -                   -
  ieee-802.1   7            class-7    -                   -
```

The following example displays information for the specified classifier.

```
supervisor@rtbrick>LEAF01: op> show qos classifier residential-pbit-classifier
Classifier: residential-pbit-classifier
Active: True
  Match Type   Codepoint    Class      Remark Codepoint    Color
  ieee-802.1   0            class-0    -                   -
  ieee-802.1   1            class-1    -                   -
  ieee-802.1   2            class-2    -                   -
  ieee-802.1   3            class-3    -                   -
  ieee-802.1   4            class-4    -                   -
  ieee-802.1   5            class-5    -                   -
  ieee-802.1   6            class-6    -                   -
  ieee-802.1   7            class-7    -                   -
```

show qos interface

Syntax:

show qos interface <option>

Option	Description
-	Without any option, this command displays a summary of all the interfaces.
<interface-name>	Displays QoS classifier information for the specified interface.

The following example displays information for the specified interface.

```
supervisor@rtbrick>LEAF01: op> show qos interface ifl-0/0/10/200
Interface           Profile
ifl-0/0/10/200      pta_8queues_comp_on_S
supervisor@rtbrick>LEAF01: op>
```

show qos policer

Syntax:

show qos policer <option>

Option	Description
-	Without any option, this command displays a summary of all the QoS policers.
<policer-name>	Displays QoS policer information.

Option	Description
counter	Displays policer counter information.
class-to-policer-map	Displays class-to-policer-map information.

The following example displays a summary of all the QoS policers.

```
supervisor@rtbrick>LEAF01: op> show qos policer
Policer: _DEFAULT_POLICER_50_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        50000          50000          33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: policer-residential
Active: True, Type: two-rate-three-color, Levels: 4, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        8000           8000           800            800            -              -
  2        -              -              -              -              -              -
  3        -              -              800            800            -              -
  4        -              -              800            800            -              -
```

The following example displays policer information for the specified policer.

```
supervisor@rtbrick>LEAF01: op> show qos policer policer-residential
Policer: policer-residential
Active: True, Type: two-rate-three-color, Levels: 4, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        8000           8000           800            800            -              -
  2        -              -              -              -              -              -
  3        -              -              800            800            -              -
  4        -              -              800            800            -              -
supervisor@rtbrick>LEAF01: op>
```

The following example displays counter information for the specified counter.

```
supervisor@rtbrick>LEAF01: op> show qos policer counter lag-1
Interface                            Level  Units      Total           Received        Dropped
lag-1                                1      Packets    15773           15773           0
                                            Bytes      2555226         2555226         0
lag-1                                2      Packets    15778           15778           0
                                            Bytes      2556036         2556036         0
lag-1                                3      Packets    15775           15775           0
                                            Bytes      2555550         2555550         0
lag-1                                4      Packets    23661           23661           0
                                            Bytes      3423166         3423166         0
lag-1-egress                         1      Packets    7889            7889            0
                                            Bytes      1420020         1420020         0
lag-1-egress                         2      Packets    7889            7889            0
                                            Bytes      1420020         1420020         0
lag-1-egress                         3      Packets    7889            7889            0
                                            Bytes      1420020         1420020         0
lag-1-egress                         4      Packets    49177           49177           0
                                            Bytes      8288974         8288974         0
```

show qos profile

Syntax:

show qos profile <option>

Option	Description
-	Without any option, this command displays a summary of all the QoS profiles.
<profile-name>	Displays QoS profile information.

The following example displays a summary of all the QoS profiles.

```
supervisor@rtbrick>LEAF01: op> show qos profile lac_4queues_4classes
Profile: lac_4queues_4classes
   Classifier: residential-pbit-classifier
   Policer: policer-residential
   Scheduler map: lac_4queues_M
   Class queue map: lac_4queues_M
   Remark map: -
   Class policer map: policer-map-l2tp
   Mulifield classifier: -
supervisor@rtbrick>LEAF01: op>
```

show qos queue

Syntax:

show qos queue <option>

Option	Description
-	Without any option, this command displays a summary of all the queues.
class-to-queue-map	Displays queue class-to-queue-map information.
counter	Displays QoS queue counter information.
counter <interface-name>	Displays QoS queue counter information for the specified interface.
<interface-name>	Displays QoS queue details for the specified interface.

The following example displays a summary of all the queues.

```
supervisor@rtbrick>LEAF01: op> show qos queue
Applied queues:
  Interface          Queue              Queue Size         Min Thres          Max Thres          Drop Prob
Shaper
  ifl-0/0/10/100     BE_S               240000             -                  -                  -
-
  ifl-0/0/10/100     LD_S               200000             -                  -                  -
shaper_LD
  ifl-0/0/10/100     LL_S               200000             -                  -                  -
shaper_LL
  ifl-0/0/10/100     VO_S               50000              -                  -                  -
shaper_VO
  ifl-0/0/10/200     BE_S               240000             -                  -                  -
-
  ifl-0/0/10/200     LD_S               200000             -                  -                  -
shaper_LD
  ifl-0/0/10/200     LL_S               200000             -                  -                  -
shaper_LL
  ifl-0/0/10/200     VO_S               50000              -                  -                  -
shaper_VO
  ifl-0/0/10/300     BE_S               240000             -                  -                  -
-
  ifl-0/0/10/300     LD_S               200000             -                  -                  -
shaper_LD
  ifl-0/0/10/300     LL_S               200000             -                  -                  -
shaper_LL
  ifl-0/0/10/300     VO_S               50000              -                  -                  -
shaper_VO
Configured queues:
  Queue                Queue Size           Min Thres            Max Thres            Drop Prob
Shaper
  BE_L                 375000               -                    -                    -                    -
  BE_M                 375000               -                    -                    -                    -
  BE_S                 240000               -                    -                    -                    -
  CO_L                 312500               -                    -                    -                    -
  CO_M                 156250               -                    -                    -                    -
  CO_S                 50000                -                    -                    -                    -
  IO_L                 312500               -                    -                    -
shaper_IO
  IO_M                 156250               -                    -                    -
shaper_IO
  IO_S                 50000                -                    -                    -
shaper_IO
  LD_L                 1250000              -                    -                    -
shaper_LD
  LD_M                 625000               -                    -                    -
shaper_LD
  LD_S                 200000               -                    -                    -
shaper_LD
  LL_L                 1250000              -                    -                    -
shaper_LL
  LL_M                 625000               -                    -                    -
shaper_LL
  LL_S                 200000               -                    -                    -
shaper_LL
  VO_L                 312500               -                    -                    -
shaper_VO
  VO_M                 156250               -                    -                    -
shaper_VO
  VO_S                 50000                -                    -                    -
shaper_VO
  free_6_L             375000               -                    -                    -                    -
  free_6_M             375000               -                    -                    -                    -
  free_6_S             240000               -                    -                    -                    -
  free_7_L             375000               -                    -                    -                    -
  free_7_M             375000               -                    -                    -                    -
  free_7_S             240000               -                    -                    -                    -
```

The following example displays queue information for the specified interface.

```
supervisor@rtbrick>LEAF01: op> show qos queue ifl-0/0/10/100
Applied queues:
  Interface          Queue              Queue Size         Min Thres          Max Thres          Drop Prob
Shaper
  ifl-0/0/10/100     BE_S               240000             -                  -                  -
-
  ifl-0/0/10/100     LD_S               200000             -                  -                  -
shaper_LD
  ifl-0/0/10/100     LL_S               200000             -                  -                  -
shaper_LL
  ifl-0/0/10/100     VO_S               50000              -                  -                  -
shaper_VO
```

The following example displays queue counter information.

```
supervisor@rtbrick>LEAF01: op> show qos queue counter
Interface                      Queue Group           Queue                  Class    Units      Received
Queued              Dropped

                                                                                     Bytes      0
0                   0
ifl-0/1/32/6                   olt-dpu-mgmt-queues    OLT_MGMT               0        Packets    0
0                   0
                                                                                     Bytes      0
0                   0
ifl-0/1/33/4                   olt-dpu-mgmt-queues    OLT_MGMT               0        Packets    0
0                   0
                                                                                     Bytes      0
0                   0
ifl-0/1/33/6                   olt-dpu-mgmt-queues    OLT_MGMT               0        Packets    0
0                   0
                                                                                     Bytes      0
0                   0
ppp-0/1/30/72339069014638594   pta-4queues           BE_PTA                 0        Packets    941111
288270              652841
                                                                                     Bytes      942823712
288837428           653986284
ppp-0/1/30/72339069014638594   pta-4queues           LD_PTA                 2        Packets    938859
446474              492385
                                                                                     Bytes      942614436
448259896           494354540
ppp-0/1/30/72339069014638594   pta-4queues           LL_PTA                 1        Packets    938851
480506              458345
                                                                                     Bytes      942606404
482428024           460178380
ppp-0/1/30/72339069014638594   pta-4queues           VO_PTA                 3        Packets    3667257
673116              2994141
                                                                                     Bytes      953486820
175010160           778476660
l2bsa-0/1/30/281479271677953   l2bsa-4queues         BE_L2BSA               0        Packets    0
0                   0
                                                                                     Bytes      0
0                   0
l2bsa-0/1/30/281479271677953   l2bsa-4queues         LD_L2BSA               2        Packets    0
0                   0
                                                                                     Bytes      0
0                   0
l2bsa-0/1/30/281479271677953   l2bsa-4queues         LL_L2BSA               1        Packets    0
0                   0
                                                                                     Bytes      0
0                   0
```

The following example displays queue counter information for the specified interface.

```
supervisor@rtbrick>LEAF01: op> show qos queue counter ifl-0/1/30/6
Interface                      Queue Group           Queue                  Class    Units      Received
Queued              Dropped
ifl-0/1/30/6                   olt-dpu-mgmt-queues    OLT_MGMT               0        Packets    0
0                   0
```

```
                                                                              Bytes      0
0                 0
```

show qos scheduler

Syntax:

show qos scheduler <option>

Option	Description
-	Without any option, this command displays a summary of all the schedulers.
<scheduler-name>	Displays scheduler information for the specified scheduler.

The following example displays a summary of all the schedulers.

```
supervisor@rtbrick>LEAF01: op> show qos scheduler
Scheduler              Type                  Shaper                Composite           Active
fff                    strict_priority       -                     False                 False
fffd                   strict_priority       -                     False                 False
lac_4queues            strict_priority       -                     True                  False
olt-pon1               fair_queueing         -                     False                 False
olt-pon10              fair_queueing         -                     False                 False
olt-pon11              fair_queueing         -                     False                 False
olt-pon12              fair_queueing         -                     False                 False
olt-pon13              fair_queueing         -                     False                 False
olt-pon14              fair_queueing         -                     False                 False
olt-pon15              fair_queueing         -                     False                 False
olt-pon16              fair_queueing         -                     False                 False
olt-pon17              fair_queueing         -                     False                 False
olt-pon18              fair_queueing         -                     False                 False
olt-pon19              fair_queueing         -                     False                 False
olt-pon2               fair_queueing         -                     False                 False
olt-pon20              fair_queueing         -                     False                 False
olt-pon21              fair_queueing         -                     False                 False
olt-pon22              fair_queueing         -                     False                 False
olt-pon23              fair_queueing         -                     False                 False
olt-pon24              fair_queueing         -                     False                 False
olt-pon25              fair_queueing         -                     False                 False
olt-pon26              fair_queueing         -                     False                 False
olt-pon27              fair_queueing         -                     False                 False
olt-pon28              fair_queueing         -                     False                 False
olt-pon29              fair_queueing         -                     False                 False
olt-pon3               fair_queueing         -                     False                 False
olt-pon30              fair_queueing         -                     False                 False
olt-pon31              fair_queueing         -                     False                 False
olt-pon32              fair_queueing         -                     False                 False
olt-pon4               fair_queueing         -                     False                 False
olt-pon5               fair_queueing         -                     False                 False
olt-pon6               fair_queueing         -                     False                 False
olt-pon7               fair_queueing         -                     False                 False
olt-pon8               fair_queueing         -                     False                 False
olt-pon9               fair_queueing         -                     False                 False
pta_4queues_comp_off   strict_priority       -                     True                  False
pta_4queues_comp_on    strict_priority       -                     True                  False
pta_8queues_comp_off   strict_priority       -                     True                  False
```

```
pta_8queues_comp_on    strict_priority        -                    True                 False
supervisor@rtbrick>LEAF01: op>
```

The following example displays scheduler information for the specified scheduler.

```
supervisor@rtbrick>LEAF01: op> show qos scheduler lac_4queues
Scheduler              Type                   Shaper               Composite           Active
lac_4queues            strict_priority        -                    True                 False
```

show qos scheduler-map

Syntax:

show qos scheduler-map <option>

Option	Description
-	Without any option, this command displays a summary of all the scheduler maps.
<scheduler-map>	Displays scheduler information for the specified scheduler map.

The following example displays a summary of all the scheduler maps.

```
supervisor@rtbrick>LEAF01: op> show qos scheduler-map
Scheduler-Map: lac_4queues_S
   Scheduler: fff                  Scheduler: strict_priority
     Queue: LD_S                   strict_priority_1
     Scheduler: pta_4queues_comp_off  Scheduler: strict_priority
       Queue: LL_S                   strict_priority_1
       Queue: VO_S                   strict_priority_0
   Scheduler: fffd                 Scheduler: strict_priority
     Queue: BE_S                   strict_priority_0
Scheduler-Map: schedmap-olt
   Scheduler: olt-pon1             Scheduler: fair_queueing
   Scheduler: olt-pon2             Scheduler: fair_queueing
   Scheduler: olt-pon3             Scheduler: fair_queueing
   Scheduler: olt-pon4             Scheduler: fair_queueing
   Scheduler: olt-pon5             Scheduler: fair_queueing
   Scheduler: olt-pon6             Scheduler: fair_queueing
   Scheduler: olt-pon7             Scheduler: fair_queueing
   Scheduler: olt-pon8             Scheduler: fair_queueing
   Scheduler: olt-pon9             Scheduler: fair_queueing
   Scheduler: olt-pon10            Scheduler: fair_queueing
   Scheduler: olt-pon11            Scheduler: fair_queueing
   Scheduler: olt-pon12            Scheduler: fair_queueing
   Scheduler: olt-pon13            Scheduler: fair_queueing
   Scheduler: olt-pon14            Scheduler: fair_queueing
   Scheduler: olt-pon15            Scheduler: fair_queueing
   Scheduler: olt-pon16            Scheduler: fair_queueing
   Scheduler: olt-pon17            Scheduler: fair_queueing
```

```
    Scheduler: olt-pon18              Scheduler: fair_queueing
    Scheduler: olt-pon19              Scheduler: fair_queueing
    Scheduler: olt-pon20              Scheduler: fair_queueing
    Scheduler: olt-pon21              Scheduler: fair_queueing
    Scheduler: olt-pon22              Scheduler: fair_queueing
    Scheduler: olt-pon23              Scheduler: fair_queueing
    Scheduler: olt-pon24              Scheduler: fair_queueing
    Scheduler: olt-pon25              Scheduler: fair_queueing
    Scheduler: olt-pon26              Scheduler: fair_queueing
    Scheduler: olt-pon27              Scheduler: fair_queueing
    Scheduler: olt-pon28              Scheduler: fair_queueing
    Scheduler: olt-pon29              Scheduler: fair_queueing
    Scheduler: olt-pon30              Scheduler: fair_queueing
    Scheduler: olt-pon31              Scheduler: fair_queueing
    Scheduler: olt-pon32              Scheduler: fair_queueing
    Scheduler: olt-pon33              Scheduler: False
 Scheduler-Map: lac_4queues_L
    Scheduler: lac_4queues            Scheduler: strict_priority
      Queue: BE_L                     strict_priority_1
      Queue: LD_L                     strict_priority_1
      Queue: LL_L                     strict_priority_2
      Queue: VO_L                     strict_priority_0
```

The following example displays scheduler information for the specified scheduler-map.

```
supervisor@rtbrick>LEAF01: op> show qos scheduler-map lac_4queues_S
Scheduler-Map: lac_4queues_S
   Scheduler: fff                   Scheduler: strict_priority
     Queue: LD_S                    strict_priority_1
     Scheduler: pta_4queues_comp_off  Scheduler: strict_priority
       Queue: LL_S                    strict_priority_1
       Queue: VO_S                    strict_priority_0
   Scheduler: fffd                  Scheduler: strict_priority
     Queue: BE_S                    strict_priority_0
```

show qos shaper

Syntax:

show qos shaper <option>

Option	Description
-	Without any option, this command displays a summary of all the shapers.
<shaper-name>	Displays scheduler information for the specified shaper.

The following example displays a summary of all the shapers.

```
supervisor@rtbrick>LEAF01: op> show qos shaper
Shaper              High Rate(Kbps)     Low Rate(Kbps)      High Burst(Kb)      Low Burst(Kb)       Active
pon-shaper          2488000             -                   -                   -                   True
shaper_IO           -                   1000000             -                   -                   True
shaper_LD           1000000             -                   -                   -                   True
shaper_LL           1000000             -                   -                   -                   True
shaper_VO           1000000             -                   -                   -                   True
shaper_session      1000000             100                 -                   -                   True
```

The following example displays shaper information for the specified shaper.

```
supervisor@rtbrick>LEAF01: op> show qos shaper shaper_session
Shaper              High Rate(Kbps)     Low Rate(Kbps)      High Burst(Kb)      Low Burst(Kb)       Active
shaper_session      1000000             100                 -                   -                   False
```

show qos multifield-classifier

Syntax:

show qos multifield-classifier <option>

Option	Description
-	Without any option, this command displays a summary of all the multifield classifiers.
<multifield-classifier-name>	Displays scheduler information for the specified multifield classifiers.

The following example displays a summary of all the multifield classifiers.

```
supervisor@rtbrick>LEAF01: op> show qos multifield-classifier
Multifield Classifier: global-mfc
  ACL type: multifield_ipv4
  Ordinal: 10000
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 224
    Action:
      Forward class: class-7
  Priority: 1000
  Ordinal: 9000
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 192
    Action:
      Remark codepoint: 184
      Forward class: class-6
  Ordinal: 6200
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 160
    Action:
      Remark codepoint: 184
      Forward class: class-5
  Ordinal: 5200
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 64
    Action:
      Remark codepoint: 184
      Forward class: class-2
  Ordinal: 6000
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 96
    Action:
      Remark codepoint: 184
      Forward class: class-3
  Ordinal: 5000
    Match:
      Source IPv4 prefix: 198.51.100.2/24
    Action:
      Remark codepoint: 184
      Forward class: class-0
  Ordinal: 6100
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 128
    Action:
      Remark codepoint: 184
      Forward class: class-4
  Ordinal: 5100
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 32
    Action:
      Remark codepoint: 184
      Forward class: class-1
  Ordinal: 100
    Match:
      Source IPv6 prefix: 2001:db8:0:100::/32
      IPv6 TC: 224
    Action:
      Remark codepoint: 196
      Forward class: class-3
Multifield Classifier: ipoe-double-play
  ACL type: multifield_ipv4
  Ordinal: 5200
    Match:
      Destination IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 160
    Action:
      Remark codepoint: 224
      Forward class: class-2
  Ordinal: 5000
    Match:
      Destination IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 64
    Action:
      Remark codepoint: 224
      Forward class: class-0
  Ordinal: 5300
    Match:
      Destination IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 192
    Action:
      Remark codepoint: 224
      Forward class: class-3
  Ordinal: 5100
    Match:
      Destination IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 96
    Action:
      Remark codepoint: 224
      Forward class: class-1
supervisor@rtbrick>LEAF01: op>
```

The following example displays multifield-classifier information for the specified multifield-classifier.

```
supervisor@rtbrick>LEAF01: op> show qos multifield-classifier global-mfc
Multifield Classifier: global-mfc
  ACL type: multifield_ipv4
  Ordinal: 10000
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 224
    Action:
      Forward class: class-7
  Priority: 1000
  Ordinal: 9000
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 192
    Action:
      Remark codepoint: 184
      Forward class: class-6
  Ordinal: 6200
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 160
    Action:
      Remark codepoint: 184
      Forward class: class-5
  Ordinal: 5200
    Match:
      Source IPv4 prefix: 198.51.100.2/24
      IPv4 TOS: 64
    Action:
      Remark codepoint: 184
      Forward class: class-2
supervisor@rtbrick>LEAF01: op>
```

Displaying QoS Statistics for Layer-2 and Layer-3 Logical Interfaces

To display QoS Statistics for layer-2 and layer-3 logical interfaces, enter the following command:

show interface <layer-2/layer-3 interface> **statistics**

To obtain detailed view of QoS statistics for layer-2 and layer-3 logical interfaces, enter the following command:

show interface <layer-2/layer-3 interface> **detail**

To clear QoS statistics of layer-2 and layer-3 logical interfaces, enter the following command:

clear interface statistics <layer-2/layer-3 interface>

The following example displays QoS Statistics for the interface ifl-0/1/46/501.

```
supervisor@rtbrick>LEAF01: op> show interface ifl-0/1/46/501 statistics
Logical Interface:  ifl-0/1/46/501
  VPP statistics:
    Counter               Unit      Count
  Packet statistics:
    Ingress forwarded packets: 67598            Ingress forwarded bytes: 69872136
    Ingress drop Packets: 12378422              Ingress drop bytes: 12799221528
    Egress forwarded packets: 67370             Egress forwarded bytes: 69369760
    Egress drop packets: 0                      Egress drop bytes: 0
    Egress Class (Queue) Statistics:
        class-0: 67370 packets 69369760 bytes dropped: 0 packets 0 bytes
        class-1: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-2: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-3: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-4: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-5: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-6: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-7: 0 packets 0 bytes dropped: 0 packets 0 bytes
    Ingress Policer Statistics:
        Level 1: 0 packets 0 bytes dropped: 0 packets 0 bytes
        Level 2: 0 packets 0 bytes dropped: 0 packets 0 bytes
        Level 3: 0 packets 0 bytes dropped: 0 packets 0 bytes
        Level 4: 67598 packets 69872136 bytes dropped: 12378422 packets 12799221528 bytes
supervisor@rtbrick>ufi07.q2c.u21.r4.nbg.rtbrick.net: op>
```

The following example displays detailed QoS Statistics for the interface ifl-0/1/46/501.

```
supervisor@rtbrick>LEAF01: op> show interface ifl-0/1/46/501 detail
Logical Interface:  ifl-0/1/46/501
  VPP statistics:
    Counter               Unit      Count
  Packet statistics:
    Ingress forwarded packets: 67598            Ingress forwarded bytes: 69872136
    Ingress drop Packets: 12378422              Ingress drop bytes: 12799221528
    Egress forwarded packets: 67370             Egress forwarded bytes: 69369760
    Egress drop packets: 0                      Egress drop bytes: 0
    Egress Class (Queue) Statistics:
        class-0: 67370 packets 69369760 bytes dropped: 0 packets 0 bytes
        class-1: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-2: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-3: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-4: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-5: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-6: 0 packets 0 bytes dropped: 0 packets 0 bytes
        class-7: 0 packets 0 bytes dropped: 0 packets 0 bytes
    Ingress Policer Statistics:
        Level 1: 0 packets 0 bytes dropped: 0 packets 0 bytes
        Level 2: 0 packets 0 bytes dropped: 0 packets 0 bytes
        Level 3: 0 packets 0 bytes dropped: 0 packets 0 bytes
        Level 4: 67598 packets 69872136 bytes dropped: 12378422 packets 12799221528 bytes
```

13.9. LLDP

13.9.1. LLDP Overview

Link Layer Discovery Protocol (LLDP) is a media-independent link layer protocol used by network devices for advertising their identity, capabilities to neighbors on a LAN segment. LLDP runs over the data-link layer only, allowing two systems running different network layer protocols to learn about each other.

LLDP supports a set of attributes that it uses to discover neighbor devices. These attributes contain type, length, and value descriptions and are referred to as TLVs. LLDP supported devices can use TLVs to receive and send information to their neighbors. Details such as configuration information, device capabilities, and device identity can be advertised using this protocol.

Guidelines

- All LLDP packets are sent to the CPU for further processing.
- LLDP packets will be lower priority packets when compared with the routing protocol packets. In case of a congestion, the LLDP packets may be policed.

Limitations

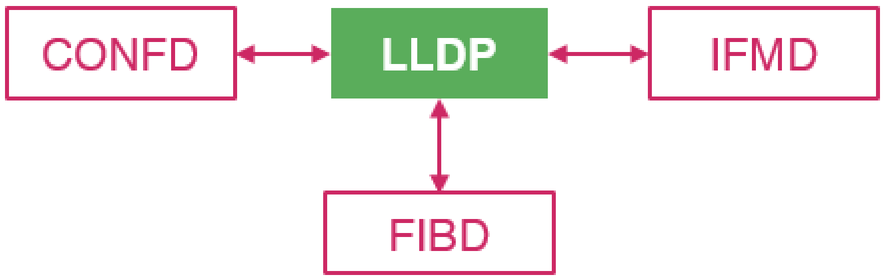
- LLDP currently does not intend to work on bundle interfaces.
- LLDP is disabled for the following interfaces on L2BSA L2X:

  Physical interfaces on Q2C platform

  LAG member interfaces on QAX platform

Interactions with other features

RBFS provides various daemons that run as background processes. LLDP interacts with these daemons for accessing necessary configurations.



The table below shows the interactions of LLDP with the other daemons of RBFS.

Daemon Name	**Daemon Description**	**Interaction Details**
Confd	Configuration	*Confd* stores all the LLDP configuration information. LLDP subscribes for the following tables of *Confd*: • global.lldp.config • global.lldp.interface.config
IFMD	Interface Monitoring	LLDP is enabled on the physical interface. LLDP subscribes to the IFP table to find the status of the physical link. An LLDP message can be sent only to those interfaces whose link state is UP.
FIBD	Forwarding Information Base	*FIBD* subscribes for the LLDP interface table which has all the information required for enabling LLDP on the interface. This is used to program in VPP. Any neighbor detected, VPP will notify and this detection is stored in the FIBD table for tracking purpose. Each time a message is received from the neighbor, it will be updated in this table. Thus, the history of the messages received from the neighbor is tracked. This information is stored in one of the FIBD tables. LLDP subscribes to this table to find its neighbor.

LLDP Limitations with L2X

LLDP is disabled in the following scenario:

1. The outgoing-interface is configured in the egress L2X.
2. The outgoing-interface is configured in the ingress L2X with the match-type as "any" or "untagged".
3. The outgoing-interface and incoming-interface are configured in the bi-directional L2X.

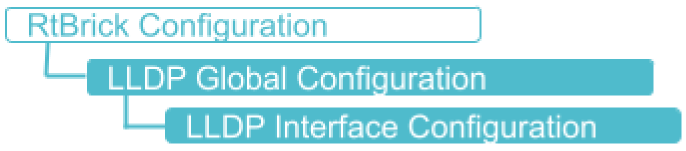
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

13.9.2. LLDP Configuration

Configuration Hierarchy

The diagram illustrates the LLDP configuration hierarchy.



Configuration Syntax and Commands

The following sections describe the LLDP configuration syntax and commands.

LLDP Global Configuration

Syntax:

set lldp <attribute> <value>

Attribute	Description
admin-status [disable|enable]	Enable or disable LLDP. LLDP is enabled, by default.
system-description <system description>	LLDP global system description to be sent to the neighbor
system-name <system name>	LLDP global system name to be sent to the neighbor. If the system name is not configured, it is fetched from BDS.
tx-hold <transmit hold time>	Specifies the amount of time (in seconds) a receiving device maintains the neighbor information before aging your device. If the timer expires and no LLPD packet was received, the neighbor will be marked as DOWN. Default value is 120 seconds. The hold-time range is 1 through 360000.
tx-interval <advertisement interval>	Interval (in seconds) at which LLDP packets are sent to neighbors. Default interval value is 30 seconds. The transmission interval range is 1 through 3600 seconds.

Example 1: Enable LLDP

```
{
    "rtbrick-config:lldp": {
      "admin-status": "enable",
      "system-name": "rtbrick",
      "system-description": "This is rtbrick system",
      "tx-interval": 40,
      "tx-hold": 150
    }
  }
```

LLDP Interface Configuration

Syntax:

set lldp interface <interface-name> <attribute> <value>

Attribute	Description
interface <interface name>	Name of the interface on which you enable or disable LLDP. By default, the interface is enabled. NOTE: Interface level configuration will override the global LLDP enable/disable functionality.
admin-status [disable or enable]	LLDP is enabled by default. The command *set lldp admin-status disable* is used to disable it. If you want to re-enable it, run the *set lldp admin-status enable* command.
port-desc <port description>	LLDP port description to be sent to the neighbor. If the port description is not configured, the description configured under interface from IFMD is taken as LLDP port description.

Example 1: LLDP on Interface Configuration

```
{
    "rtbrick-config:lldp": {
      "interface": [
        {
          "interface-name": "ifp-0/0/1",
          "port-description": "this is port ifp-0/0/1",
          "admin-status": "enable"
        }
      ]
    }
  }
```

13.9.3. LLDP Operational Commands

LLDP Show Commands

The LLDP show commands provide detailed information about the LLDP global summary, neighbors, and interfaces.

Syntax:

show lldp <option>

Option	Description
interface	Displays information about interfaces where LLDP is enabled.
neighbor	Displays information about all the neighbors.
summary	Displays global summary details such as system name, frequency of transmissions, the hold-time for the packets sent, TLVs, and the disabled TLVs.

LLDP Interfaces

The command displays the information about interfaces where LLDP is enabled.

Syntax:

show lldp interface <attribute> <value>

Option	Description
interface-name	Displays summary details such as status, MAC address and description.

Example 1: LLDP interface summary

```
root@rtbrick: op> show lldp interface
Interface name       Status  MAC address           Description
ifp-0/0/0                   Down    7a:77:6a:01:00:01   Physical interface #0 from
node 0, chip 0
ifp-0/0/1                   Up         7a:77:6a:01:00:02   Physical interface #1
from node 0, chip 0
ifp-0/0/2                   Up          7a:77:6a:01:00:05   Physical interface #2
from node 0, chip 0
```

Example 2: For the interfaces where L2X is configured, LLDP is suppressed as shown in the following example.

```
root@rtbrick: op> show lldp interface
Interface name Status MAC address Description
ifp-0/0/0 Suppress 7a:3a:ce:60:00:00 Suppressed by L2X : L2X1
ifp-0/0/1 Suppress 7a:3a:ce:60:00:01 Suppressed by L2X : L2X2
```

LLDP Neighbors

This command displays the information of all neighbors.

Syntax:

show lldp neighbor <attribute> <value>

Option	Description
detail	Displays the information in detail about a specific LLDP neighbor or all neighbors.
interface-name	Name of the interface on which neighbor is formed.

Example 1: LLDP neighbor summary

```
root@rtbrick: op> show lldp  neighbor
Neighbor name      Status  Remote port ID     Local port ID      Neighbor MAC
address  Last received     Last sent
fwdd-r2                    Up         ifp-0/0/1                 ifp-0/0/1
7a:1a:c9:00:00:01             0:00:05 ago       0:00:05 ago
fwdd-r2                    Up         ifp-0/0/2                 ifp-0/0/2
7a:1a:c9:00:00:04             0:00:05 ago       0:00:05 ago
```

Example 2: LLDP all neighbor details

```
root@rtbrick: op> show lldp neighbor detail

Neighbor: fwdd-r2
  Neighbor MAC address: 7a:1a:c9:00:00:01
  Neighbor port ID: ifp-0/0/1
  Neighbor port description: Physical interface #1 from node 0, chip 0
  Neighbor TTL: 121
  Neighbor timeout: 121000
  Local interface: ifp-0/0/1
  Local MAC address: 7a:77:6a:01:00:02
  Local port description: Physical interface #1 from node 0, chip 0
  Packets sent: 35
  Packets received: 36
  Neighbor status: Up

Neighbor: fwdd-r2
  Neighbor MAC address: 7a:1a:c9:00:00:04
  Neighbor port ID: ifp-0/0/2
  Neighbor port description: Physical interface #2 from node 0, chip 0
  Neighbor TTL: 121
  Neighbor timeout: 121000
  Local interface: ifp-0/0/2
  Local MAC address: 7a:77:6a:01:00:05
  Local port description: Physical interface #2 from node 0, chip 0
  Packets sent: 35
  Packets received: 36
  Neighbor status: Up
```

Example 3: LLDP specific neighbor details

```
root@rtbrick: op> show lldp neighbor ifp-0/0/1
Neighbor: fwdd-r2
  Neighbor MAC address: 7a:1a:c9:00:00:01
  Neighbor port ID: ifp-0/0/1
  Neighbor port description: Physical interface #1 from node 0, chip 0
  Neighbor TTL: 121
  Neighbor timeout: 121000
  Local interface: ifp-0/0/1
  Local MAC address: 7a:77:6a:01:00:02
  Local port description: Physical interface #1 from node 0, chip 0
  Packets sent: 148
  Packets received: 149
  Neighbor status: Up
```

To access the Operational State API that corresponds to this CLI, click here.

LLDP Summary

The command displays the LLDP global summary information.

Syntax:

show lldp summary

Option	Description
-	Without any options, the command displays LLDP global summary details such as status, MAC address and description.

Example 1: LLDP System Summary

```
root@rtbrick: op> show lldp summary
Mode: global
  System hostname: fwdd-r1
  Transmit interval: 30 sec
  Transmit holdtime: 120 sec
```

LLDP Clear Commands

Clear commands allow to reset operational states.

LLDP Neighbor

This commands resets LLDP neighbor.

Syntax:

clear lldp neighbor <option>

Option	Description
all	Clears all the LLDP neighbors.
<ifp-0/0/1>	Clears a specific neighbor on an interface.

Example: The example below shows how to clear all the LLDP neighbors.

```
supervisor@rtbrick: op> clear lldp neighbor all
```

14. Multicast

14.1. IGMP

14.1.1. IGMP Overview

Internet Group Management (IGMP) protocol allows a host to advertise its multicast group membership to neighboring switches and routers. IGMP is a standard protocol used by the TCP/IP protocol suite to achieve dynamic multicasting.

There are two components in the IGMP solution:

- IGMPv2/v3 Client: It sends Join or Leave messages to a multicast group. Typical example of a client is a SET-TOP box. The IGMP client can respond to any IGMP general queries or group-specific queries that are received.
- Multicast Router: The recipient of IGMP Join/Leave message. After receiving the message, it determines whether the corresponding message needs to be processed or not. After processing the IGMP messages, it sends this information to its multicast upstream router. Along with this, it can program certain entries in its routers which results in forwarding specific multicast packets on that interface.

IGMPv3 Lite

IGMP version 3 adds support for "source filtering", that is, the ability for a system to report interest in receiving packets **only** from specific source addresses, or from **all but** specific source addresses, sent to a particular multicast address. That information may be used by multicast routing protocols to avoid delivering multicast packets from specific sources to networks where there are no interested receivers.

The RtBrick IGMP v3lite solution adds support for source filtering. Source filtering enables a multicast receiver host to signal from which groups it wants to receive multicast traffic, and from which sources this traffic is expected. That information may be used by multicast routing protocols to avoid delivering multicast packets from specific sources to networks where there are no interested receivers.

IGMP Version 3 will help conserve bandwidth by allowing a host to select the specific sources from which it wants to receive traffic. Also, multicast routing protocols will be able to make use of this information to conserve bandwidth when constructing the branches of their multicast delivery trees.

Static Joins

After an interface on a multicast device is configured to statically join an IGMP group, the multicast device considers that the interface has static multicast group members and sends multicast packets to this interface, regardless of whether hosts connected to this interface request the multicast packets.

SSM Mapping

SSM mapping takes IGMPv2 reports and converts them to IGMPv3. In case of legacy devices, there could be a possibility that BNG might receive IGMPv2 membership reports. If BNG receives an IGMPv2 membership for a specific group G1, BNG uses the SSM mapping configuration to determine one or more Source (S) addresses for a given group. This SSM mappings are translated to the IGMPv3 joins like IGMPV3 JOIN INCLUDE (G, [S1, G1], [S2, G1] and so on) and BNG continues to process as if it has received from the subscriber.

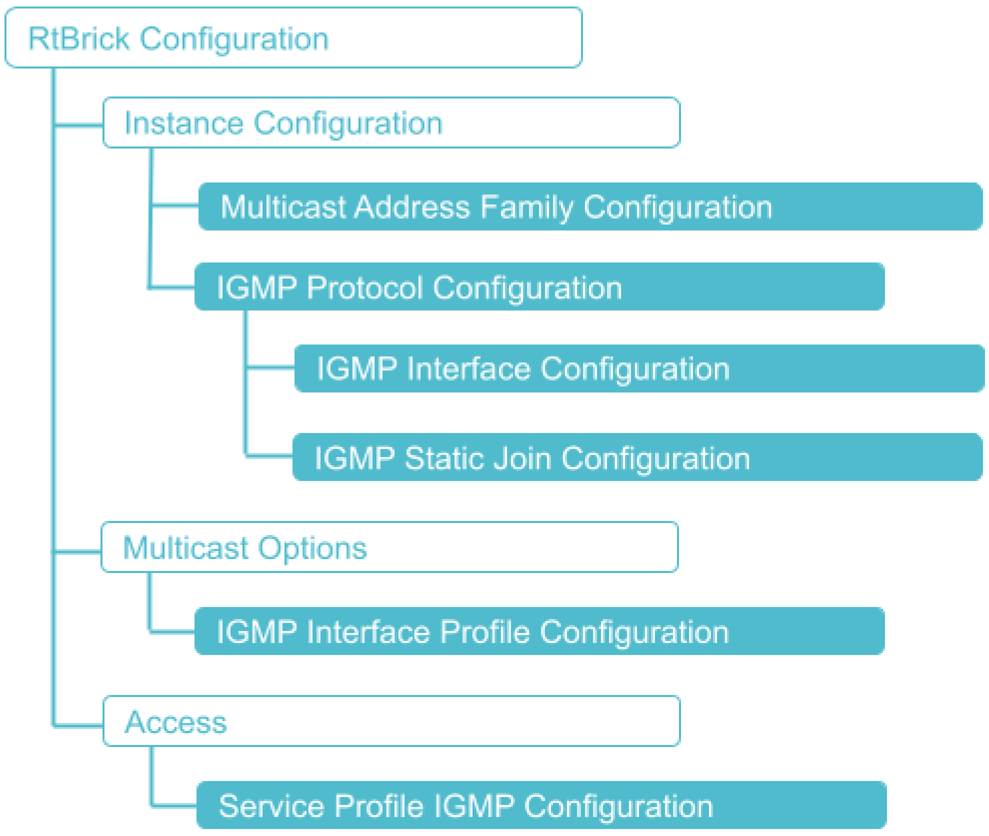
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

14.1.2. IGMP Configuration

Configuration Hierarchy

The diagram illustrates the IGMP configuration hierarchy.


RtBrick Configuration
Instance Configuration
Multicast Address Family Configuration
IGMP Protocol Configuration
IGMP Interface Configuration
IGMP Static Join Configuration
Multicast Options
IGMP Interface Profile Configuration
Access
Service Profile IGMP Configuration

Configuration Syntax and Commands

The following sections describe the interface configuration syntax and commands.

Multicast Address Family Configuration

You can enable the multicast IPv4 address family under the IGMP instance using the following command:

Syntax:

set instance <instance> **address-family** <attribute> <value>

Attribute	Description
<instance>	Specifies the name of the network instance
<afi>	Address family identifier (AFI). Supported value: ipv4
<safi>	Subsequent address family identifier (SAFI), that is, multicast.

Example: Multicast Address Family Configuration

```
{
    "rtbrick-config:address-family": [
      {
        "afi": "ipv4",
        "safi": "multicast"
      }
    ]
  }
```

IGMP Protocol Configuration

To configure an IGMP on an instance, the same instance should be enabled globally with AFI IPv4 and SAFI as both unicast and multicast.

Syntax:

set instance <instance> **protocol igmp** <attribute> <values>

If no instance is specified, IGMP will be enabled on the default instance.

Options	Description
<instance>	Name of the IGMP instance.
interfaces <…>	IGMP interface configuration. Refer to section 2.2.1.1 for the IGMP interface configuration.
robustness-variable <variable-value>	The robustness value is used by IGMP to determine the number of times to send messages. Default value: 3. Range: 0-255.
source-address <source-address>	Source address of the IGMP query at the instance-level. NOTE: IF subscriber IFL is configured with the source address, then takes priority; otherwise, the the instance-level source address will be used. If source address is not configured, 0.0.0.0 will be the default address.
static-group <…>	Static multicast route configuration. Refer to section 2.2.2.2 for the IGMP static join configuration.

Example: IGMP Configuration

```
{
    "rtbrick-config:igmp": {
      "robustness-variable": 5,
      "source-address": "198.51.100.177"
    }
  }
```

IGMP Interface Configuration

When you start IGMP on an interface, it operates with the default settings.

Syntax:

set instance <instance> **protocol igmp interfaces interface** <interface-name> <attribute> <value>

Attribute	Description
<instance>	Name of the instance
<interface-name>	Name of the IP multicast interface
max-groups <count>	Specifies the maximum count of multicast group memberships
version <version>	Specifies the IGMP version, that is, IGMPv2 or IGMPv3
interface-profile <profile>	Name of the interface configuration profile

Example: IGMP Interface Configuration

```
{
    "rtbrick-config:interface": [
      {
        "interface-name": "ifl-0/0/0/1",
        "version": "IGMPv3",
        "max-groups": 30,
        "interface-profile": "profile1"
      }
    ]
  }
```

IGMP Static Join Configuration

Syntax:

set instance <instance> **protocol igmp static-group** <attribute> <value>

Command Parameters

<instance>	Specifies the instance name
<group-address>	Specifies the multicast address
<outgoing-interface>	Name of the outbound interface. The null0 is a discard or sink interface for IGMP static join configuration. If IGMP Static Join is configured with the Null0 outgoing interface, then it is mandatory to configure the null0 IGMP interface.

Example: IGMP Static Join Configuration

```
{
    "rtbrick-config:static-group": [
      {
        "group-address": "198.51.100.200",
        "source-address": "198.51.100.1",
        "outgoing-interface": "null0"
      }
    ]
  }
```

IGMP Interface Profile Configuration

Syntax:

set multicast-options igmp interface-profile <attribute> <value>

Attribute	**Description**
filter-policy <filter-policy>	Specifies the filter policy. The policy should be defined under policy statement.
immediate-leave <enable | disable>	Enable or disable the immediate leave option. The immediate-leave attribute removes group membership immediately upon receiving a group leave membership report. If enabled, IGMP perform an immediate leave upon receiving an IGMP group leave message. If the router is IGMP-enabled, it sends an IGMP last member query with a last member query response time. However, the router does not wait for the response time before it prunes off the group querier-timeout-interval IGMP other querier timeout. Default: 425s

Attribute	Description
query-interval <query-interval>	IGMP query interval in seconds. The query interval ranges from 1 to 1024 seconds. The default value is 125 seconds.
query-max-response-time <query-max-response-time>	Maximum query response interval in seconds. The maximum query response interval ranges from 1 to 1024 seconds. The default value is 100 seconds.
ssm-map-policy <ssm-map-policy>	IGMP SSM policy name. The policy for (**,G) mapping to (S,G**)
start-query-count <start-query-count>	Specifies the number of queries sent out on startup, separated by the Start Query Interval. The start query count ranges from 1 to 1024. The default value is 3.
start-query-interval <start-query-interval>	Specifies the start query interval. The start-query-interval ranges from 1 to 1024 seconds. The default value is 31 seconds (query-interval/4).

Example: IGMP Interface Profile Configuration

```
{
    "rtbrick-config:interface-profile": [
      {
        "profile-name": "profile1",
        "immediate-leave": "enable",
        "query-interval": 30,
        "query-max-response-time": 10,
        "start-query-count": 10,
        "start-query-interval": 10,
        "filter-policy": "filter_policy",
        "ssm-map-policy": "ssm_policy"
      }
    ]
  }
```

Service Profile IGMP Configuration

Syntax:

set access service-profile <profile-name> **igmp** <attribute> <value>

Attribute	Description
<profile-name>	Name of the service profile

Attribute	Description
enable <true|false>	Enable IGMP service
max-members <max-members>	Maximum IGMP membership per subscriber
profile <profile>	IGMP profile
version [IGMPv1/IGMPv2/IGMPv3]	IGMP version. The default IGMP version is IGMPv3.

Example: Service Profile IGMP Configuration

```
{
    "rtbrick-config:service-profile": [
      {
        "profile-name": "service-profile1",
        "igmp": {
          "enable": "true",
          "profile": "INTERFACE_PROFILE_1",
          "version": "IGMPv3",
          "max-members": 10
        }
      }
    ]
  }
```

IGMP Configuration Example

```
{
  "ietf-restconf:data": {
    "rtbrick-config:instance": [
      {
        "name": "default",
        "protocol": {
          "igmp": {
            "robustness-variable": 5,
            "source-address": "198.51.100.91"
          }
        }
      }
    ],
    "rtbrick-config:multicast-options": {
      "igmp": {
        "interface-profile": [
          {
            "profile-name": "INTERFACE_PROFILE_1",
            "query-interval": 10,
            "filter-policy": "FILTER_POLICY_1"
          },
          {
            "profile-name": "INTERFACE_PROFILE_2",
            "query-interval": 20,
            "ssm-map-policy": "SSM_POLICY_1"
          }
        ]
      }
    },
    "rtbrick-config:policy": {
      "statement": [
        {
          "name": "FILTER_POLICY_1",
          "ordinal": [
            {
              "ordinal": 1,
              "match": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "ipv4-mcast-group",
                    "value-type": "discrete",
                    "match-type": "or-longer",
                    "value": "198.51.100.20/24"
                  }
                ]
              },
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "operation": "return-deny"
                  }
                ]
              }
            },
            {
              "ordinal": 2,
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "operation": "return-permit"
                  }
                ]
              }
            }
          ]
        },
        {
          "name": "SSM_POLICY_1",
          "ordinal": [
            {
              "ordinal": 1,
              "match": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "ipv4-mcast-group",
                    "value-type": "discrete",
                    "match-type": "or-longer",
                    "value": "198.51.100.10/24"
                  }
                ]
              },
              "action": {
                "rule": [
                  {
                    "rule": 1,
                    "type": "ipv4-mcast-source",
                    "operation": "overwrite",
                    "value": "198.51.100.11/24"
                  }
                ]
              }
            }
          ]
        }
      ]
    },
    "rtbrick-config:access": {
      "interface": {
        "double-tagged": [
          {
            "interface-name": "ifp-0/0/1",
            "outer-vlan-min": 1,
            "outer-vlan-max": 4049,
            "inner-vlan-min": 1,
            "inner-vlan-max": 4049,
            "access-type": "PPPoE",
            "access-profile-name": "access-profile1",
            "service-profile-name": "service-profile1",
            "aaa-profile-name": "aaa-profile1"
          }
        ]
      },
      "service-profile": [
        {
          "profile-name": "service-profile1",
          "igmp": {
            "enable": "true",
            "profile": "INTERFACE_PROFILE_1",
            "version": "IGMPv3",
            "max-members": 10
          }
        }
      ]
    }
  }
}
```

14.1.3. IGMP Operational Commands

IGMP Show Commands

Syntax:

show igmp <option>

Option	Description
group	IGMP group summary information
group <group>	IGMP group detailed information
group instance <name>	IGMP group summary information in a specific instance
group outgoing-interface <interface_name>	IGMP group detailed information over a specific interface
interface	IGMP logical-interface summary information
interface instance <name>	IGMP interface summary information on specific instance
interface <interface_name>	IGMP interface detailed information

Example 1: Display IGMP interface details for all instances

```
supervisor@rtbrick>LEAF01: op> show igmp interface
Interface                        Primary Address   State          Querier Address
Instance          Uptime
null0                            n/a               n/a            n/a
vpn1            n/a
ppp-0/0/3/72339069014638597      198.51.100.100     Querier        198.51.100.133
vpn1    03h:37m:31s
```

Example 2: Display the interface summary for a specific instance

```
supervisor@rtbrick>LEAF01: op> show igmp interface instance vpn1
Interface                        Primary Address   State          Querier Address
Instance          Uptime
null0                            n/a               n/a            n/a
vpn1            n/a
ppp-0/0/3/72339069014638597      198.51.100.100     Querier       198.51.100.133
vpn1            03h:37m:39s
```

Example 3: Display IGMP group summary on all instances

```
supervisor@rtbrick>LEAF01: op> show igmp group
Source Address      Group Address       Interface                    Instance
Uptime         Expires    Version
198.51.100.79      198.51.100.233       null0                          vpn1
03h:42m:33s    n/a         IGMP
198.51.100.43   198.51.100.222        null0                          vpn1
03h:42m:33s    n/a         IGMP
198.51.100.51     198.51.100.71         ppp-0/0/3/72339069014638597     vpn1
00h:32m:58s    1m 43s      IGMPv3
```

```
198.51.100.51      198.51.100.72        ppp-0/0/3/72339069014638597     vpn1
00h:33m:03s    1m 48s      IGMPv3
198.51.100.53         198.51.100.73     ppp-0/0/3/72339069014638597     vpn1
00h:35m:26s    3m 36s      IGMPv3
198.51.100.54         198.51.100.74     ppp-0/0/3/72339069014638597     vpn1
00h:35m:26s    3m 35s      IGMPv3
198.51.100.56         198.51.100.115    ppp-0/0/3/72339069014638597     vpn1
03h:38m:16s    3m 33s      IGMPv3
198.51.100.57         198.51.100.117    ppp-0/0/3/72339069014638597     vpn1
03h:38m:16s    3m 29s      IGMPv3
198.51.100.58         198.51.100.18     ppp-0/0/3/72339069014638597     vpn1
03h:38m:16s    3m 42s      IGMPv3
198.51.100.59         198.51.100.19     ppp-0/0/3/72339069014638597     vpn1
03h:38m:16s    3m 35s      IGMPv3
198.51.100.90        198.51.100.64      ppp-0/0/3/72339069014638597     vpn1
03h:38m:16s    3m 40s      IGMPv3
198.51.100.225       198.51.100.68      ppp-0/0/3/72339069014638597     vpn1
00h:35m:26s    3m 40s      IGMPv3
```

Example 4: Display the group summary on specific instance

```
supervisor@rtbrick>LEAF01: op> show igmp group instance vpn1
Source Address      Group Address       Interface                     Instance
Uptime         Expires     Version
198.51.100.79     198.51.100.233        null0                           vpn1
03h:42m:37s    n/a         IGMP
198.51.100.43   198.51.100.233          null0                           vpn1
03h:42m:37s    n/a         IGMP
198.51.100.51       198.51.100.71       ppp-0/0/3/72339069014638597     vpn1
00h:33m:02s    1m 40s      IGMPv3
198.51.100.51       198.51.100.72       ppp-0/0/3/72339069014638597     vpn1
00h:33m:07s    1m 45s      IGMPv3
198.51.100.53       198.51.100.73       ppp-0/0/3/72339069014638597     vpn1
00h:35m:30s    3m 41s      IGMPv3
198.51.100.54       198.51.100.74       ppp-0/0/3/72339069014638597     vpn1
00h:35m:30s    3m 43s      IGMPv3
198.51.100.56       198.51.100.115      ppp-0/0/3/72339069014638597     vpn1
03h:38m:20s    3m 43s      IGMPv3
198.51.100.57       198.51.100.117      ppp-0/0/3/72339069014638597     vpn1
03h:38m:20s    3m 42s      IGMPv3
198.51.100.58        198.51.100.18      ppp-0/0/3/72339069014638597     vpn1
03h:38m:20s    3m 39s      IGMPv3
198.51.100.59         198.51.100.19     ppp-0/0/3/72339069014638597     vpn1
03h:38m:20s    3m 42s      IGMPv3
198.51.100.90        198.51.100.64      ppp-0/0/3/72339069014638597     vpn1
03h:38m:20s    3m 37s      IGMPv3
198.51.100.225       198.51.100.68      ppp-0/0/3/72339069014638597     vpn1
00h:35m:30s    3m 36s      IGMPv3
```

Example 5: Display detailed group information for specific group and source on all instances

```
supervisor@rtbrick>LEAF01: op> show igmp group 198.51.100.233 198.51.100.79
(198.51.100.79, 198.51.100.233)
  Outgoing interface      : null0
  Instance                : vpn1
  Source                  : Static
  State                   : No Members Present
  Version                 : IGMP
  Uptime                  : 03h:42m:54s
  Expires                 : n/a
  Membership interval     : n/a
  Last reporter           : n/a
  Last member query count : n/a
  Last member interval    : n/a
  Retransmit time         : n/a
  Max response time       : n/a
```

Example 6: Display detailed group information for specific group and source on all instances

```
supervisor@rtbrick>LEAF01: op> show igmp group outgoing-interface ppp-
0/0/3/72339069014638597
(198.51.100.51, 198.51.100.71)
  Outgoing interface      : ppp-0/0/3/72339069014638597
  Instance                : vpn1
  Source                  : Dynamic
  State                   : Members Present
  Version                 : IGMPv3
  Uptime                  : 00h:33m:44s
  Expires                 : 1m 43s
  Membership interval     : 110s
  Last reporter           : 198.51.100.100
  Last member query count : 3
  Last member interval    : 1s
  Retransmit time         : 1s
  Max response time       : 0s
(198.51.100.51, 198.51.100.72)
  Outgoing interface      : ppp-0/0/3/72339069014638597
  Instance                : vpn1
  Source                  : Dynamic
  State                   : Members Present
  Version                 : IGMPv3
  Uptime                  : 00h:33m:49s
  Expires                 : 1m 48s
  Membership interval     : 110s
  Last reporter           : 198.51.100.100
  Last member query count : 3
  Last member interval    : 1s
  Retransmit time         : 1s
  Max response time       : 0s
(198.51.100.53, 198.51.100.73)
  Outgoing interface      : ppp-0/0/3/72339069014638597
  Instance                : vpn1
  Source                  : Dynamic
  State                   : Members Present
  Version                 : IGMPv3
  Uptime                  : 00h:36m:12s
  Expires                 : 3m 37s
  Membership interval     : 225s
  Last reporter           : 198.51.100.100
  Last member query count : 3
  Last member interval    : 1s
  Retransmit time         : 1s
  Max response time       : 0s
(198.51.100.54, 198.51.100.74)
  Outgoing interface      : ppp-0/0/3/72339069014638597
  Instance                : vpn1
  Source                  : Dynamic
  State                   : Members Present
  Version                 : IGMPv3
  Uptime                  : 00h:36m:12s
  Expires                 : 3m 41s
  Membership interval     : 225s
  Last reporter           : 198.51.100.100
  Last member query count : 3
  Last member interval    : 1s
  Retransmit time         : 1s
  Max response time       : 0s
(198.51.100.56, 198.51.100.115)
  Outgoing interface      : ppp-0/0/3/72339069014638597
  Instance                : vpn1
  Source                  : Dynamic
  State                   : Members Present
  Version                 : IGMPv3
  Uptime                  : 03h:39m:02s
  Expires                 : 3m 38s
  Membership interval     : 225s
  Last reporter           : 198.51.100.100
  Last member query count : 3
  Last member interval    : 1s
  Retransmit time         : 1s
  Max response time       : 0s
```

Example 7: Display detailed group information for specific group and source on selected instance

```
supervisor@rtbrick>LEAF01: op> show igmp group instance vpn1 198.51.100.117
198.51.100.57
(198.51.100.57, 198.51.100.117)
  Outgoing interface      : ppp-0/0/3/72339069014638597
  Instance                : vpn1
  Source                  : Dynamic
  State                   : Members Present
  Version                 : IGMPv3
  Uptime                  : 03h:39m:31s
  Expires                 : 3m 34s
  Membership interval     : 225s
  Last reporter           : 198.51.100.100
  Last member query count : 3
  Last member interval    : 1s
  Retransmit time         : 1s
  Max response time       : 0s
supervisor@rtbrick>LEAF01: op>
```

IGMP Clear Commands

IGMP Interface

Syntax:

clear igmp interface <attribute> <value>

Option	Description
<interface_name>	Clears the specified IGMP interface
instance <instance> statistics	Clears the IGMP interface statistics for the specified instance.

Example: Clear interface IGMP statistics

```
supervisor@rtbrick>LEAF01: op> clear igmp interface instance default statistics
Interface IGMP statistics were successfully cleared
supervisor@rtbrick>LEAF01: op>
```

IGMP Group

Syntax:

clear igmp group <attribute> <value>

Option	Description
all	Clear all IGMP groups present on all instances
instance <instance>	Clear all IGMP groups present on specific instance
interface <interface_name>	Clear all IGMP groups present on specific interface

Example: Clear all IGMP groups present on all instances

```
supervisor@rtbrick>LEAF01: op> clear igmp group all
IGMP groups were successfully cleared
supervisor@rtbrick>LEAF01: op>
```

14.2. PIM

14.2.1. PIM Overview

c Protocol Independent Multicast (PIM) is a multicast routing protocol that runs over an existing unicast infrastructure. RBFS supports PIM source-specific multicast (SSM). PIM-SSM uses a subset of PIM sparse mode and IGMP version 3 (IGMPv3) to permit a client to receive multicast traffic directly from the source.

PIM SSM builds shortest-path trees (SPTs) rooted at the source immediately because in SSM, the router closest to the interested receiver host is informed of the unicast IP address of the source for the multicast traffic. That is, PIM SSM bypasses the RP connection stage through shared distribution trees, as in PIM sparse mode, and goes directly to the source-based distribution tree.

Internet Protocol Television (IPTV) is a service where digital TV signal data is delivered by using Internet protocol (IP). IPTV service networks typically use PIM-SSM as the multicast routing protocol which has the following characteristics:

- Source specific host membership report for a particular multicast group. IGMPv3 allows a host to describe specific sources from which it would like to receive data.
- PIM shortest path forwarding. Source-specific host report for a particular multicast group and initiating PIM (S,G) joins directly and immediately as result.
- No shared tree forwarding. In order to achieve global effectiveness of SSM, all networks must agree to restrict data forwarding to source trees for some recognized group range. The address range 232.0.0.0/8 has been allocated by IANA for use by source specific multicast (SSM).

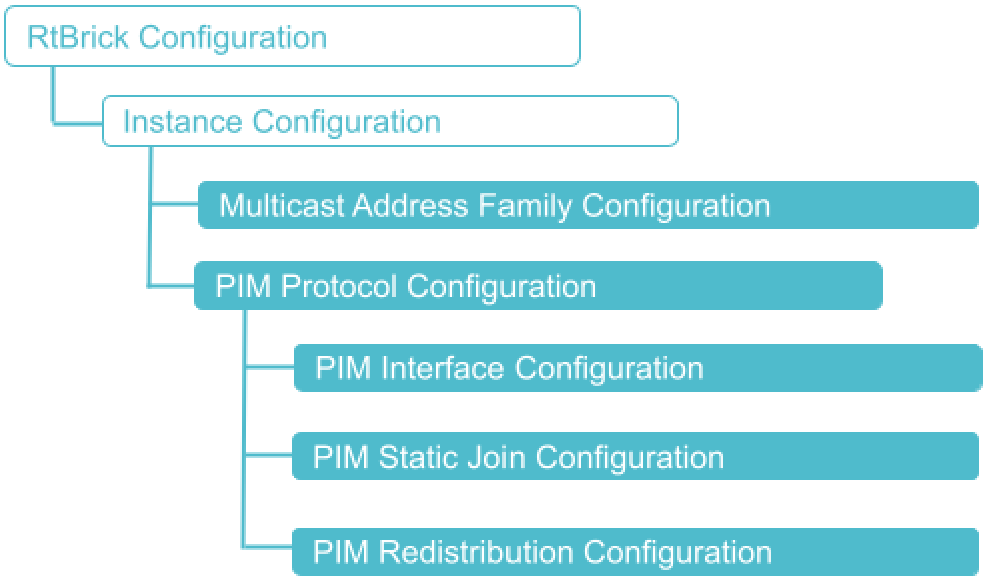
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

14.2.2. PIM Configuration

Configuration Hierarchy

The diagram illustrates the PIM configuration hierarchy.


RtBrick Configuration
Instance Configuration
Multicast Address Family Configuration
PIM Protocol Configuration
PIM Interface Configuration
PIM Static Join Configuration
PIM Redistribution Configuration

Configuration Syntax and Commands

The following sections describe the interface configuration syntax and commands.

Multicast Address Family Configuration

You can enable the multicast IPv4 address family under the PIM instance using the following command:

Syntax:

set instance <instance> **address-family** <attribute> <value>

Attribute	Description
<instance>	Specifies the name of the network instance
<afi>	Address family identifier (AFI). Supported values: ipv4
<safi>	Subsequent address family identifier (SAFI), that is, multicast.

Example: Multicast Address Family Configuration

```
{
    "rtbrick-config:address-family": [
      {
        "afi": "ipv4",
        "safi": "multicast"
      }
    ]
```

```
  }
```

PIM Protocol Configuration

Syntax:

set instance <instance> **protocol pim** <attribute> <value>

If no instance is specified, PIM will be enabled on the default instance. RBFS supports only IPV4 address family.

Attribute	Description
<instance>	Specifies the name of the instance
afi <afi>	Address family identifier (AFI). Supported: ipv4
join-prune-interval <join-prune-interval>	PIM join & prune interval. The interval ranges from 10 to 600 seconds. By default, join & prune interval is 60 seconds and hold-down timer is 210 seconds.
sparse-mode interface <… >	Reference to an entry in the global interface list. Refer to section 2.2.1.1 for the PIM interface configuration.
sparse-mode static-join <…>	A static pim join, (*,G) or (S,G). Refer to section 2.2.1.2 for the PIM static join configuration.
sparse-mode redistribute <…>	Redistribution-related configuration. Refer to section 2.2.1.3 for the PIM redistribution configuration.

PIM Interface Configuration

Syntax:

set instance <instance> **protocol pim sparse-mode interface** <attribute> <value>

Attribute	Description
<instance>	Specifies the name of the instance
<interface-name>	Reference to an entry in the global interface list
afi <afi>	Address family identifier (AFI). Supported: ipv4
dr-priority <dr-priority>	Specifies the Specifies the designated router (DR) priority value.

Attribute	Description
hello-interval <hello-interval>	Specifies the hello timer in seconds. The hello timer ranges from 1 to 3600 seconds. Default: 30 seconds.
override-interval <override-interval>	Specifies the override interval in milliseconds. Default: 2000 milliseconds.
propagation-delay <propagation-delay>	Specifies the propagation delay in milliseconds. Default: 500 milliseconds.

Example: PIM Interface Configuration

```
{
    "rtbrick-config:interface": [
      {
        "interface-name": "ifl-0/0/0/1",
        "hello-interval": 100,
        "dr-priority": 101
      },
      {
        "interface-name": "ifl-0/0/1/2"
      },
      {
        "interface-name": "ifl-0/0/3/3",
        "propagation-delay": 103,
        "override-interval": 1000
      }
    ]
  }
```

PIM Static Join Configuration

Syntax:

set instance <instance> **protocol pim sparse-mode static-join** <attribute> <value>

Attribute	Description
<instance>	Specifies the name of the instance
<interface-name>	Reference to an entry in the global interface list
<outgoing-interface>	Multicast outbound interface name
<group-address>	Multicast group IP address
<source-address>	Multicast source IP address

Example: PIM Static Join Configuration

```
{
    "rtbrick-config:static-join": [
      {
        "outgoing-interface": "ifl-0/0/0/1",
        "group-address": "198.51.100.12",
        "source-address": "198.51.100.120"
      }
    ]
  }
```

PIM Redistribution Configuration

Syntax:

set instance <instance> **protocol pim sparse-mode redistribute** <attribute> <value>

Command Parameters

<instance>	Specifies the name of the instance.
<afi>	Specifies the address family identifier. Supported value: ipv4.
<safi>	Specifies the subsequent address family identifier. Supported value: multicast.
<source>	Source protocol from which routes are being redistributed such as BGP or static.

Example: PIM Redistribution Configuration

```
{
    "rtbrick-config:redistribute": [
      {
        "afi": "ipv4",
        "safi": "multicast",
        "source": "bgp"
      }
    ]
  }
```

PIM Configuration Example

```
{
    "rtbrick-config:pim": {
      "sparse-mode": {
        "interface": [
          {
            "interface-name": "ifp-0/0/1"
          }
        ],
        "redistribute": [
          {
            "afi": "ipv4",
            "safi": "multicast",
            "source": "bgp"
          }
        ],
        "static-join": [
          {
            "outgoing-interface": "null0",
            "group-address": "198.51.100.110",
            "source-address": "198.18.73.250"
          },
          {
            "outgoing-interface": "null0",
            "group-address": "198.51.100.111",
            "source-address": "198.18.73.250"
          },
          {
            "outgoing-interface": "null0",
            "group-address": "198.51.100.112",
            "source-address": "198.18.73.250"
          },
          {
            "outgoing-interface": "null0",
            "group-address": "198.51.100.113",
            "source-address": "198.18.73.250"
          }
        ]
      }
    }
  }
```

14.2.3. PIM Operational Commands

PIM Show Commands

PIM Interface

Syntax:

show pim interface <option>

Option	Description
-	Without any option, the commands displays the information for all interfaces.

Option	Description
<interface_name>	Displays the PIM interface detail for the default instance
instance <instance_name>	Displays interface summary on specific instance

Example 1: Summary of PIM logical interface for all instances.

```
supervisor@rtbrick: op> show pim interface
Interface       Instance      IP Address        State      DR                Generator ID
ifl-1/7/1/1     default       198.51.100.247    Non-DR     198.51.100.42     1896236448
null0           default       n/a               n/a        n/a               n/a
ifl-0/0/3/3     vpn1          198.51.100.25     Non-DR     198.51.100.33     2123016228
```

Example 2: Summary of interfaces for the specified instance.

```
supervisor@rtbrick: op> show pim interface instance vpn1
Interface       Instance      IP Address        State      DR                Generator ID
ifl-0/0/3/3     vpn1          198.51.100.25     Non-DR     198.51.100.33     2123016228
```

Example 3: Detailed view of the specified PIM interface.

```
supervisor@rtbrick: op> show pim interface ifl-0/0/3/3
Interface: ifl-0/0/3/3
  Instance                   : vpn1
  State                      : Non-DR
  Primary address            : 198.51.100.25
  Generation ID              : 2123016228
  Timer values
    Hello interval           : 35s
    Join/Prune interval      : 35s
    Hold interval            : 105s
    Override interval        : 2000ms
    Prune delay interval     : 500ms
  DR election
    DR address               : 198.51.100.33
    DR priority              : 1
    DR election count        : 251
  Negotiated
    DR priority used         : True
    Lan delay used           : False
    Lan prune interval       : 0
    Lan override  used       : False
    Lan override interval    : 0
  Statistics
    Hello
      Received               : 17214
      Sent                   : 95
    Membership
      Received               : 0
      Sent                   : 759
    Assert
```

```
       Received               : 0
       Sent                   : 0
```

To access the Operational State API that corresponds to this CLI, click here.

PIM Membership

Syntax:

show pim membership <option>

Option	Description
-	Without any option, the commands displays PIM membership summary on all instances.
instance <instance_name>	Displays the PIM membership summary information on specific instance.
detail	Detailed information on all BGP peers in all instances in a list view.
instance <instance_name> detail	Displays the PIM membership detailed information on specific instance.
<group_address>	Specifies the multicast multicast group address
<source_address>	Specifies the source from which the multicast traffic is received

Example 1: Summary of the PIM membership on all instances.

```
supervisor@rtbrick: op> show pim membership
Source                   Group              Interface     Instance        Uptime
198.51.100.42            198.51.100.222     null0         default         00h:38m:05s
198.51.100.42            198.51.100.187     null0         default         00h:38m:05s
198.51.100.42            198.51.100.235     null0         default         00h:38m:05s
```

Example 2: Summary of the PIM membership for the specified instance.

```
supervisor@rtbrick: op> show pim membership instance default
Source                   Group              Interface      Instance        Uptime
198.51.100.42            198.51.100.222     null0          default         00h:38m:18s
198.51.100.42            198.51.100.187     null0          default         00h:38m:18s
```

```
198.51.100.42          198.51.100.235    null0          default        00h:38m:18s
```

Example 3: Detailed view of PIM membership on all instances.

```
supervisor@rtbrick: op> show pim membership detail
198.51.100.42, 198.51.100.222
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:29s
198.51.100.42, 198.51.100.187
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:29s
198.51.100.42, 198.51.100.235
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:29s
```

Example 4: Detailed view of PIM membership for the specified instance.

```
supervisor@rtbrick: op> show pim membership instance default detail
198.51.100.42, 198.51.100.222
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:39s
198.51.100.42, 198.51.100.187
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:39s
198.51.100.42, 198.51.100.235
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:39s
```

Example 5: Detailed view of PIM membership for the specified group-address and source-address in selected instance.

```
supervisor@rtbrick: op> show pim membership instance default 198.51.100.222 198.51.100.42
198.51.100.42, 198.51.100.222
  Instance           : default
  Outgoing interface : null0
  Source             : pim
  Subtype            : Join
  Subsource          : Static
  Uptime             : 00h:39m:50s
supervisor@rtbrick: op>
```

PIM Join and Prune

Syntax:

show pim join-prune <option>

Option	Description
-	Without any option, the commands displays join and prune summary command on all instances.
instance <instance_name>	Displays join and prune summary on a specific instance.
detail	Displays PIM join and prune detailed information on all instances.
instance <instance_name> detail	Displays detailed join and prune information in selected instance.
<group_address>	Displays PIM join and prune detailed information for a specific group on all instance
<source_address>	Specifies the source from which the multicast traffic is received.

Example 1: Summary of the PIM join and prune all instances.

```
supervisor@rtbrick: op> show pim join-prune
Source                 Group             Upstream Interface    Instance
198.51.100.42          198.51.100.222     ifl-1/7/1/1          default
198.51.100.42          198.51.100.187     ifl-1/7/1/1          default
198.51.100.42          198.51.100.235     ifl-1/7/1/1          default
198.51.100.11          198.51.100.22      ifl-0/0/3/3          vpn1
198.51.100.11          198.51.100.217     ifl-0/0/3/3          vpn1
```

Example 2: Summary of the PIM join and prune summary for the specified instance.

```
supervisor@rtbrick: op> show pim join-prune instance vpn1
Source            Group              Upstream Interface  Instance
198.51.100.11     198.51.100.22      ifl-0/0/3/3         vpn1
198.51.100.11     198.51.100.217     ifl-0/0/3/3         vpn1
198.51.100.33     198.51.100.23      ifl-0/0/3/3         vpn1
198.51.100.40     198.51.100.24      ifl-0/0/3/3         vpn1
```

Example 3: Detailed view of the PIM join and prune for the specified instance.

```
supervisor@rtbrick: op> show pim join-prune instance vpn1 detail
198.51.100.11, 198.51.100.22
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.11
  Type               : Join
  Uptime             : 00h:01m:11s
198.51.100.11, 198.51.100.217
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.11
  Type               : Join
  Uptime             : 00h:01m:11s
198.51.100.33, 198.51.100.23
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.33
  Type               : Join
  Uptime             : 00h:03m:41s
198.51.100.40, 198.51.100.24
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.40
  Type               : Join
  Uptime             : 00h:03m:41s
198.51.100.66, 198.51.100.196
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.66
  Type               : Join
  Uptime             : 00h:55m:31s
198.51.100.70, 198.51.100.197
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.70
  Type               : Join
  Uptime             : 00h:55m:30s
198.51.100.80, 198.51.100.198
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.80
  Type               : Join
  Uptime             : 00h:55m:30s
198.51.100.38, 198.51.100.199
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.38
  Type               : Join
  Uptime             : 00h:55m:31s
```

```
198.51.100.200, 198.51.100.210
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.200
  Type               : Join
  Uptime             : 00h:55m:31s
```

Example 4: Detailed view of the PIM join and prune for the specified instance, source, and group.

```
supervisor@rtbrick: op> show pim join-prune instance vpn1 198.51.100.210 198.51.100.200
198.51.100.200, 198.51.100.210
  Instance           : vpn1
  Upstream interface : ifl-0/0/3/3
  Upstream neighbor  : 198.51.100.200
  Type               : Join
  Uptime             : 00h:55m:50s
```

PIM Neighbors

Syntax:

show pim neighbor <option>

Option	Description
-	Without any option, the commands displays the PIM neighbor summary on all instances.
instance <instance_name>	Displays the PIM neighbor summary information on specific instance
instance <instance_name> <neighbor_address>	Displays detailed information for specific PIM neighbor in selected instance.

Example 1: Summary of PIM neighbor on all instances.

```
supervisor@rtbrick: op> show pim neighbor
Neighbor          Interface         Instance     Generation ID  Uptime         Expires
198.51.100.42     ifl-1/7/1/1       default      1413290566     00h:55m:34s    1m 35s
198.51.100.11     ifl-0/0/3/3       vpn1         666648646      00h:55m:59s    21s
198.51.100.37     ifl-0/0/3/3       vpn1         1893441310     00h:55m:59s    28s
198.51.100.33     ifl-0/0/3/3       vpn1         1582670973     00h:55m:56s    24s
198.51.100.40     ifl-0/0/3/3       vpn1         2114142516     00h:55m:59s    21s
198.51.100.50     ifl-0/0/3/3       vpn1         620803409      00h:55m:56s    29s
```

Example 2: Summary of PIM neighbor for the specified instance.

```
supervisor@rtbrick: op> show pim neighbor instance default
Neighbor          Interface          Instance       Generation ID  Uptime         Expires
198.51.100.42      ifl-1/7/1/1        default         1413290566   00h:55m:41s    1m 28s
```

Example 3: Detailed view of PIM neighbor for the specified instance.

```
supervisor@rtbrick: op> show pim neighbor instance default 198.51.100.42

Neighbor: 198.51.100.42
  Interface            : ifl-1/7/1/1
  Instance             : default
  Hold down interval   : 105s
  Expires              : 105s
  Generation ID        : 1413290566
  DR priority          : 1
  Uptime               : 00h:55m:47s
  Last transition time : Tue Nov 24 06:47:08 GMT +0000 2020
  Holddown received    : 1
```

To access the Operational State API that corresponds to this CLI, click here.

PIM Reverse Path Forwarding (RPF)

Syntax:

show pim rpf <option>

Option	Description
-	Without any option, the commands displays the PIM rpf summary information on all instance.
instance <instance_name>	Displays the PIM rpf summary information on specific instance.
instance <instance> <source_address>	Displays the PIM rpf detailed information for specific source-address in selected instance.

Example 1: Summary of PIM rpf on all instances.

```
supervisor@rtbrick: op> show pim rpf
Multicast Source  Instance       RPF Interface       Neighbor
198.51.100.42      default        ifl-1/7/1/1      198.51.100.42
198.51.100.11      vpn1           ifl-0/0/3/3      198.51.100.11
198.51.100.33      vpn1           ifl-0/0/3/3      198.51.100.33
```

```
198.51.100.40      vpn1           ifl-0/0/3/3      198.51.100.40
198.51.100.66      vpn1           ifl-0/0/3/3      198.51.100.66
198.51.100.107     vpn1           n/a              n/a
```

Example 2: Summary of PIM rpf for the specified instance.

```
supervisor@rtbrick: op> show pim rpf instance vpn1
Multicast Source  Instance       RPF Interface       Neighbor
198.51.100.11     vpn1         ifl-0/0/3/3      198.51.100.11
198.51.100.33     vpn1         ifl-0/0/3/3      198.51.100.33
198.51.100.40     vpn1         ifl-0/0/3/3      198.51.100.40
198.51.100.66     vpn1         ifl-0/0/3/3      198.51.100.66
198.51.100.70     vpn1         ifl-0/0/3/3      198.51.100.70
198.51.100.107    vpn1         n/a                 n/a
```

Example 3: Detailed view of PIM rpf for the specified source-address in selected instance.

```
supervisor@rtbrick: op> show pim rpf instance vpn1 198.51.100.11
Multicast source : 198.51.100.11
  Instance        : vpn1
  AFI             : ipv4
  SAFI            : unicast
  RPF interface   : ifl-0/0/3/3
  Peer            : 198.51.100.11
  Covering prefix : n/a
  MAC address     : 00:12:01:00:00:01
```

PIM Routes

Syntax:

show pim mroute <option>

Option	Description
-	Without any option, the commands displays the PIM routes summary information on all instance.
detail	Displays the PIM routes detail for all instances.
instance <instance_name>	Displays the PIM routes summary on specific instance
instance <instance_name> detail	Displays the PIM routes detailed information on specific instances.

Example 1: Summary of PIM routes for all instances.

```
supervisor@rtbrick: op> show pim mroute
Instance: default, AFI: ipv4, SAFI: multicast
  Source              Group               Route Source   Preference  Nexthop             OIF
  198.51.100.42      198.51.100.222   pim            240         n/a                 null0
  198.51.100.42      198.51.100.187  pim            240         n/a                 null0
  198.51.100.42      198.51.100.235  pim            240         n/a                 null0
```

Example 2: Detailed view of PIM routes for all instances.

```
supervisor@rtbrick: op> show pim mroute detail
198.51.100.42, 198.51.100.222
  Source        : pim                    Preference       : 240
  Sub source    : Static                 Subtype          : Join
  RPF neighbor : 198.51.100.42          RPF interface    : ifl-1/7/1/1
  Nexthop       : n/a                    Egress interface : null0
  Nexthop type : Multicast Fanout       NextHop action   : None
  Destination   : default-ipv4-multicast
  Resolved in   : default-ipv4-multicast
198.51.100.42, 198.51.100.187
  Source        : pim                    Preference       : 240
  Sub source    : Static                 Subtype          : Join
  RPF neighbor : 198.51.100.42          RPF interface    : ifl-1/7/1/1
  Nexthop       : n/a                    Egress interface : null0
  Nexthop type : Multicast Fanout       NextHop action   : None
  Destination   : default-ipv4-multicast
  Resolved in   : default-ipv4-multicast
198.51.100.42, 198.51.100.235
  Source        : pim                    Preference       : 240
  Sub source    : Static                 Subtype          : Join
  RPF neighbor : 198.51.100.42          RPF interface    : ifl-1/7/1/1
  Nexthop       : n/a                    Egress interface : null0
  Nexthop type : Multicast Fanout       NextHop action   : None
  Destination   : default-ipv4-multicast
  Resolved in   : default-ipv4-multicast
```

Example 3: Summary of PIM routes for the specific instance.

```
supervisor@rtbrick: op> show pim mroute instance default
Instance: default, AFI: ipv4, SAFI: multicast
  Source              Group               Route Source   Preference  Nexthop             OIF
  198.51.100.96    198.51.100.222   pim            240         n/a                 null0
  198.51.100.96    198.51.100.187   pim            240         n/a                 null0
  198.51.100.96    198.51.100.235   pim            240         n/a                 null0
```

Example 4: Detailed view of PIM routes for the specified instances.

```
supervisor@rtbrick: op> show pim mroute instance default detail
198.51.100.96, 198.51.100.222
  Source        : pim                    Preference       : 240
  Sub source    : Static                 Subtype          : Join
  RPF neighbor : 198.51.100.96          RPF interface    : ifl-1/7/1/1
  Nexthop       : n/a                    Egress interface : null0
  Nexthop type : Multicast Fanout       NextHop action   : None
  Destination   : default-ipv4-multicast
  Resolved in  : default-ipv4-multicast
198.51.100.96, 198.51.100.187
  Source       : pim                    Preference       : 240
  Sub source   : Static                 Subtype          : Join
  RPF neighbor : 198.51.100.96          RPF interface    : ifl-1/7/1/1
  Nexthop      : n/a                    Egress interface : null0
  Nexthop type : Multicast Fanout       NextHop action   : None
  Destination  : default-ipv4-multicast
  Resolved in  : default-ipv4-multicast
198.51.100.96, 198.51.100.235
  Source       : pim                    Preference       : 240
  Sub source   : Static                 Subtype          : Join
  RPF neighbor : 198.51.100.96          RPF interface    : ifl-1/7/1/1
  Nexthop      : n/a                    Egress interface : null0
  Nexthop type : Multicast Fanout       NextHop action   : None
  Destination  : default-ipv4-multicast
  Resolved in  : default-ipv4-multicast
```

To access the Operational State API that corresponds to this CLI, click here.

PIM Clear Commands

Clear commands allow to reset operational states.

PIM Neighbor

Syntax:

clear pim neighbor

Option	Description
neighbor	Clears all neighbors.

Example: The example below shows how to clear pim neighbor.

```
supervisor@rtbrick: op> clear pim neighbor
All instance clear triggered
```

PIM Neighbor Instance

Syntax:

clear pim neighbor instance <option> ...

Option	Description
neighbor instance <instance_name>	Clears all neighbors on the specified instance.
neighbor instance <instance_name> interface <interface-name>	Clears all neighbors on the specified interface.

Example 1: The example below shows how to clear neighbors of the specified instance

```
supervisor@rtbrick: op> clear pim neighbor instance ip2vrf
Instance clear triggered
```

Example 2: The example below shows how to clear neighbors of the specified interface

```
supervisor@rtbrick: op> clear pim neighbor instance ip2vrf interface ifl-0/0/0
Instance clear triggered
```

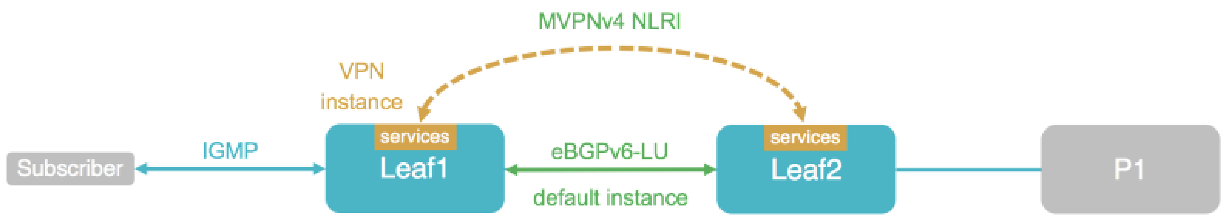
14.3. Multicast VPN

14.3.1. MVPN Overview

The Multicast VPN (MVPN) feature provides the ability to support multicast over a Layer 3 VPN. Multicast allows the efficient distribution of information between a single multicast source and multiple receivers. IP multicast is used to stream video, voice, and data to an MPLS VPN network core. The RBFS MVPN implementation is based on RFC 6513 “Multicast in MPLS/BGP IP VPNs” and RFC 6514 “BGP Encodings and Procedures for Multicast in MPLS/BGP IP VPNs”.

RFC and draft compliance are partial except as specified.

RBFS can operate in a spine/leaf fabric as shown in the diagram. The leaf device delivers access services to subscribers or assets, and the spine device provides connectivity to the core network. In such a scenario, IPv4 multicast traffic is carried in a multicast VPN instance. This also allows to deliver IPv4 multicast traffic across an IPv6-only fabric. In terms of MVPN, both leaf and spine devices act as Provider Edge (PE) routers.



Multicast subscribers will typically join multicast streams via IGMP. Leaf switches will translate and signal the join messages to the spines using BGP MVPN routes. Spine switches will further forward the join messages towards the source to the upstream core router.

Multicast Address Families

In MVPN, there are two types of instances that address families involved:

- Within the multicast VPN instance, joins are represented as multicast routes. These are AFI 1 (IPv4), SAFI 2 (Multicast). The IPv4 multicast address family needs to be enabled in the VPN service instance as well as for BGP in the VPN instance.
- In the Multicast VPN itself, messages like joins or active sources are advertised using BGP MVPN routes. These are AFI 1 (IPv4), SAFI 5 (MVPN). The BGP peerings that carry the MVPN routes typically run in the default instance. Therefore the MVPN address family needs to be enabled in the default instance, both for the BGP instance itself as well as for the BGP peerings.

In summary, the following address families need to be enabled by configuration:

- The IPv4 multicast address family in the VPN service instance.
- The IPv4 multicast address family for BGP in the VPN service instance.
- The Multicast VPN address family in the BGP default instance.
- The Multicast VPN address family in the BGP peerings in the default instance.

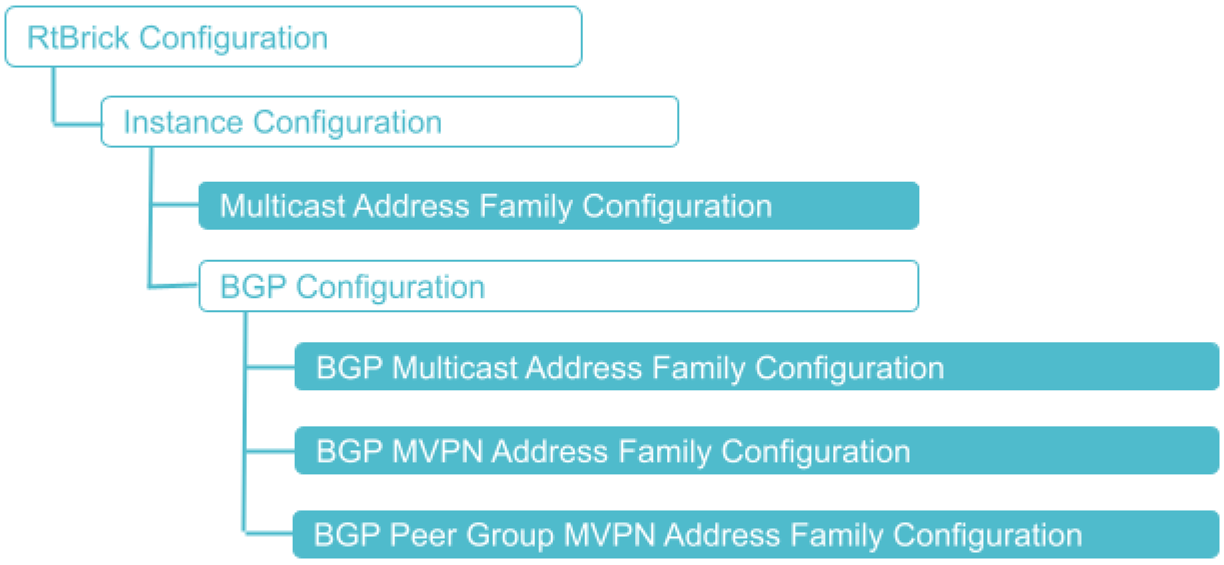
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

14.3.2. MVPN Configuration

Configuration Hierarchy

The diagram illustrates the Multicast VPN configuration hierarchy.


RtBrick Configuration
Instance Configuration
Multicast Address Family Configuration
BGP Configuration
BGP Multicast Address Family Configuration
BGP MVPN Address Family Configuration
BGP Peer Group MVPN Address Family Configuration

Configuration Syntax and Commands

The following sections describe the MVPN configuration syntax and commands.

Multicast Address Family Configuration

Enable the IPv4 multicast address family in the VPN service instance.

Syntax:

set instance <instance> **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<instance>	The name of the VPN service instance
<afi>	Address family identifier (AFI). For IPv4 multicast, the required value is: ipv4
<safi>	Subsequent address family identifier (SAFI). For IPv4 multicast, the required value is: multicast

Example: Instance Address Family Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:instance": [
      {
        "name": "services",
        "address-family": [
          {
            "afi": "ipv4",
            "safi": "multicast",
            "route-target": {
              "import": "target:198.51.100.10:13",
              "export": "target:198.51.100.10:13"
            }
          }
        ]
      }
    ]
  }
}
```

BGP Multicast Address Family Configuration

Enable the IPv4 multicast address family for BGP in the VPN service instance.

Syntax:

set instance <instance> protocol bgp **address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<instance>	The name of the VPN service instance
<afi>	Address family identifier (AFI). For IPv4 multicast, the required value is: ipv4
<safi>	Subsequent address family identifier (SAFI). For IPv4 multicast, the required value is: multicast
redistribute <source>	Optionally redistribute PIM routes into BGP to translate and advertise subscriber join message as MVPN routes. The required source value is pim as IGMP routes are handled by the PIM process too.

Example: Instance BGP Address Family Configuration

```
{
  "ietf-restconf:data": {
  "rtbrick-config:instance": {
    "name": "services",
    "protocol": {
      "bgp": {
        "address-family": [
          {
            "afi": "ipv4",
            "safi": "multicast",
            "redistribute": [
              {
                "source": "pim"
              }
            ]
          }
        ]
      }
    }
  }
}
```

BGP MVPN Address Family Configuration

Enable the MVPN address family in the global BGP instance.

Syntax:

set instance <instance> **protocol bgp address-family** <afi> <safi> <attribute> <value>

Attribute	Description
<instance>	Name of the global BGP instance. Typically this will be the default instance.
<afi>	Address family identifier (AFI). For MVPN, the required value is: ipv4
<safi>	Supported SAFIs are unicast. For MVPN, the required value is: vpn-multicast

Example: BGP MVPN Address Family Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:instance": [
      {
        "name": "default",
        "protocol": {
          "bgp": {
            "address-family": [
              {
                "afi": "ipv4",
                "safi": "vpn-multicast"
              }
              ]
            }
          }
        }
      ]
    }
  }
```

BGP Peer Group MVPN Address Family Configuration

Enable the MVPN address family for the global BGP peerings carrying the MVPN routes.

Syntax:

set instance <instance> **protocol bgp peer-group** <attribute> <value>

Attribute	Description
<instance>	Name of the global BGP instance. Typically this will be the default instance.
<pg-name>	Peer group name
address-family <afi> <safi>	BGP peer group address family specific. For MVPN, the required AFI/SAFI values are: ipv4/ vpn-multicast.

Example: BGP Peer Group MVPN Address Family Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:peer-group": [
      {
        "pg-name": "spine",
        "address-family": [
          {
            "afi": "ipv4",
            "safi": "vpn-multicast",
            "extended-nexthop": "true",
            "update-nexthop": {
              "ipv6-address": "2001:db8:0:103::"
            }
          }
        ]
      }
    ]
  }
```

Multicast VPN Configuration Example

```
{
  "ietf-restconf:data": {
    "rtbrick-config:instance": [
      {
        "name": "default",
        "protocol": {
          "bgp": {
            "address-family": [
              {
                "afi": "ipv4",
                "safi": "vpn-multicast"
              }
            ],
            "peer-group": [
              {
                "pg-name": "spine",
                "address-family": [
                  {
                    "afi": "ipv4",
                    "safi": "vpn-multicast",
                    "extended-nexthop": "true",
                    "update-nexthop": {
                      "ipv6-address": "2001:db8:0:103::"
                    }
                  }
                ]
              }
            ]
          }
        }
      },
      {
        "name": "services",
        "address-family": [
          {
            "afi": "ipv4",
            "safi": "multicast",
            "route-target": {
              "import": "target:198.51.100.10:13",
              "export": "target:198.51.100.10:13"
            }
          }
        ],
        "protocol": {
          "bgp": {
            "address-family": [
              {
                "afi": "ipv4",
                "safi": "multicast",
                "redistribute": [
                  {
                    "source": "pim"
                  }
                ]
              }
            ]
          }
        }
        }
      ]
    }
  }
```

14.3.3. MVPN Operational Commands

Show Commands

Syntax:

show mroute ipv4 instance <vpn-instance-name> <attribute> <value>

Option	Description
-	Without any option, the commands displays summary of all the multicast routes.
<vpn-instance-name>	Name of the VPN instance
detail	Displays detailed view of all the multicast routes.
group <group-address>	Multicast group address
source <source-address>	Multicast source address

Example: Display Multicast Route Summary Information

```
supervisor@rtbrick: op> show mroute ipv4 instance services group 198.51.100.100/24
detail
Instance: services, AFI: ipv4, SAFI: multicast
198.51.100.250/24, 198.51.100.100/24
  Source: pim, Preference: 250
      Next Hop type: Multicast Fanout, Next Hop action: None
      Resolved in: services-ipv4-multicast
      Egress interface: null0
```

MVPN Route Show Commands

Syntax:

show bgp fib ipv4 vpn-multicast instance default <attribute> <value>

Option	Description
-	Without any option, the commands displays summary of all the multicast routes.

Option	Description
detail	Displays detailed view of all the VPN multicast routes.
<prefix>	Destination prefix address

Example: Display MVPN Route Summary Information

```
supervisor@rtbrick: op> show bgp fib ipv4 vpn-multicast instance default
Instance: default, AFI: ipv4, SAFI: vpn-multicast
  Group            Source         Preference      Out Label            Route Type
Next Hop
  -                -                     20       20003,bos:1          Intra-AS_I-
PMSI_AD   -
  -                -                     20       20003,bos:1          Intra-AS_I-
PMSI_AD   -
  198.51.100.111/24    198.51.100.2/24  20       20003,bos:1
Source_Tree_Join     -
```

15. Security

15.1. Securing Management Plane

15.1.1. Securing the Management Plane Overview

The Securing Management Plane feature provides the capability to restrict the access to the management plane only to authenticated and authorized subjects.

The authentication identifies a subject, and the authorization validates if the subject is allowed to execute the action.

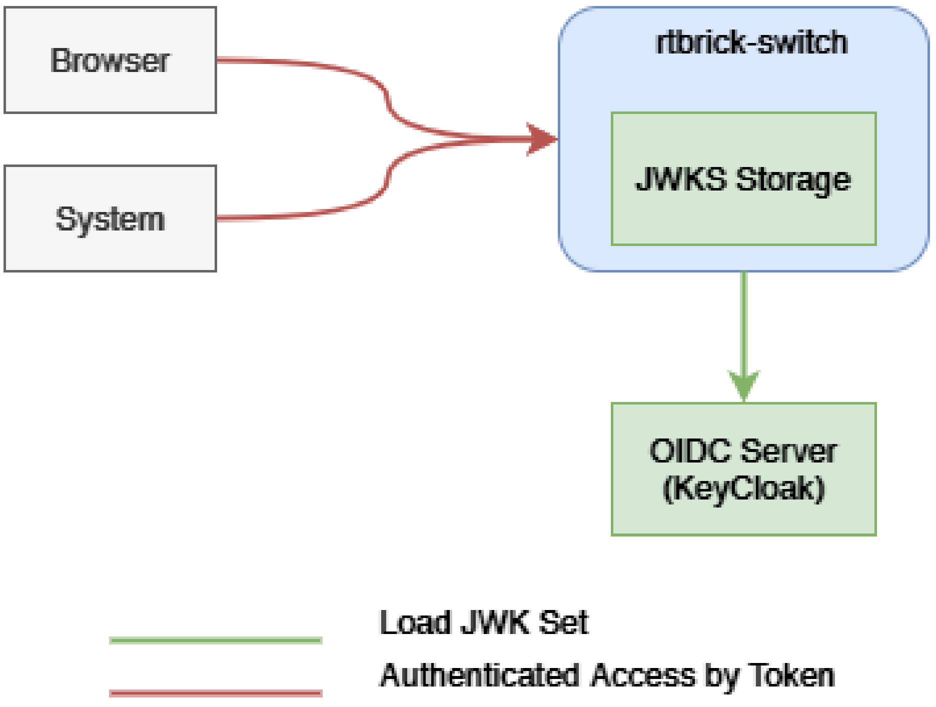


Figure 25. External Dataflow

The figure-1 shows the data flow when accessing an rtbrick-switch. Each call against the switch in more detail against the API Gateway Daemon (APIGWD) of the switch has to be authenticated with an access token. There is only one exception when accessing the CTLRD's UI; it is possible to be redirected to an OpenIDConnect Authenticator.

The APIGWD validates the access token against an JSON Web Key Set (JWKS) (https://tools.ietf.org/html/rfc7517). This key set can be loaded from a file locally on the system or auto discovered via the OpenIDConnect server.

A valid access token, in the sense of syntactically correct but also successfully validated signature by one of the JSON Web Key of the JWKS files, leads in an authenticated user. If the validation is unsuccessful, the call will be rejected.

The access token contains scopes which are used internally for the authorization checks. The authorization is a role based authorization where the scopes equal to the roles.

Internally the access token is converted to an RtBrick token, and all the communications inside the switch is authenticated via this RtBrick token.

The dataflow inside of the switch can be seen in Figure 2.

The scopes of the access token are copied to the RtBrick Token.

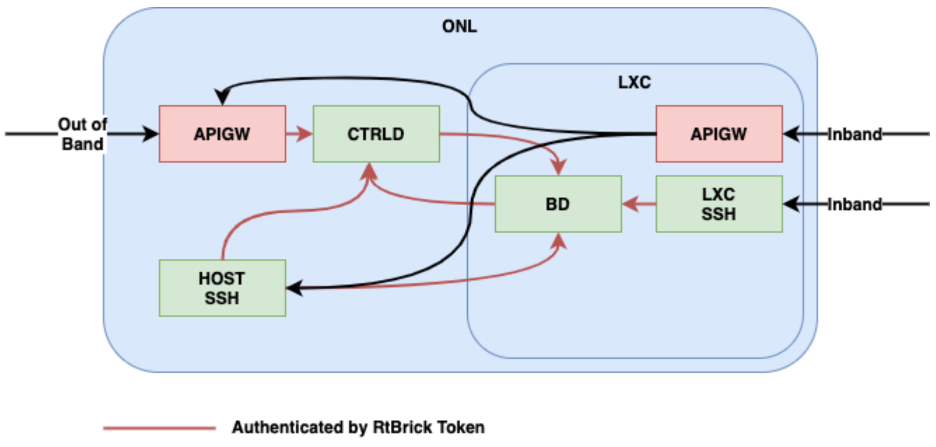


Figure 26. Internal Dataflow

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

15.1.2. RtBrick Token

JSON Web Tokens

JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object.

This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA.

RFC and draft compliance are partial except as specified.

For more information about JSON Web Token, see https://jwt.io/introduction/.

Structure

In its compact form, JSON Web Tokens consist of three parts separated by dots (.), which are:

- Header
- Payload
- Signature

Therefore, a JWT typically looks like the following.

```
xxxxx.yyyyy.zzzzz
```

Header

The header typically consists of two parts:

- The **type of the token**, which is JWT
- The signing algorithm that is being used, such as HMAC SHA256 or RSA

The suite of specifications on JWT provisions a few different options to identify particular cryptographic keys. The most straightforward mechanism is the "kid" claim. This claim can be added to the header of the token. It is intended to contain a string-based key identifier.

For example:

```
{
  "alg": "HS256",
  "typ": "JWT",
  "kid": "0815"
}
```

Then, this JSON is Base64Url encoded to form the first part of the JWT.

Payload

The second part of the token is the **payload**, which contains the claims. Claims are statements about an entity (typically, the user) and additional data.

There are three types of claims:

- registered
- public
- private

Registered claims
These are a set of predefined claims which are not mandatory but recommended, to provide a set of useful, interoperable claims. Some of them are: iss (issuer), exp (expiration time), sub (subject), aud (audience), and others.

Public claims
These can be defined at will by those using JWTs. But to avoid collisions they should be defined in the IANA JSON Web Token Registry or be defined as a URI that contains a collision resistant namespace.

Private claims
These are the custom claims created to share information between parties that agree on using them and are neither registered or public claims.

An example payload is as follows:

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022,
  "exp": 1600000000,
  "scope": "user"
}
```

The payload is then Base64Url encoded to form the second part of the JSON Web Token.

Signature

To create the signature part you have to take the encoded header, the encoded payload, a secret, the algorithm specified in the header, and sign that.

For example if you want to use the HMAC SHA256 algorithm, the signature will be created in the following way:

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

The signature is used to verify the message wasn’t changed along the way, and, in the case of tokens signed with a private key, it can also verify that the sender of the JWT is who it says it is.

Putting all together

The output is three Base64-URL strings separated by dots that can be easily passed in HTML and HTTP environments.

AccessToken

The Access Token is a JSON Web Token. The token is typically sent in the Authorization header using the Bearer schema. The content of the header should look like the following:

```
Authorization: Bearer <token>
```

The API Gateway also supports sending the token as Cookie, but this is not described here, that is only used for the CTRLD web UI.

The token has to have the kid claim in the header. This kid is used to find the right JSON Web Key (JWK) from one of the JSON Web Key Sets (JWKS).

APIGWD searches for the jwks file under /etc/rtbrick/apigwd/access_secret_jwks.json, but it is also possible to provide an additional oicd endpoint. By that the keysets are searched in the provided order:

- local file specified by command line -access-token-jwks-file-name
- oicd auto discovery -oidc-issuer

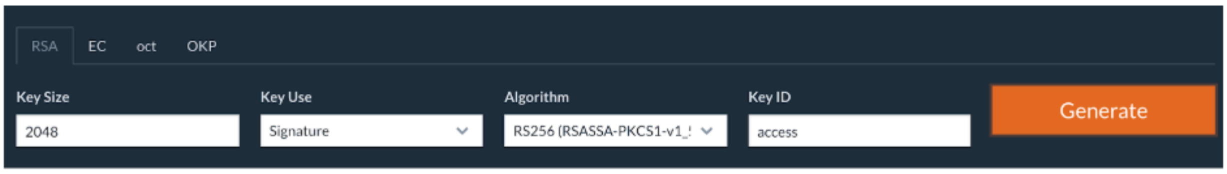
The scope claim contains the roles the user has. For example:

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true,
  "iat": 1516239022,
  "exp": 1600000000,
  "scope": "user operator"
}
```

This user has the roles user and operator.

JWKS Validation

The example below shows how to create a public/private key set file (for example with https://mkjwk.org/)



Key Size: 2048
Key Use: Signature
Algorithm: RS256 (RSASSA-PKCS1-v1_!
Key ID: access

Example PubPrivJwks.json:

```
{
  "keys": [
    {
      "p": "-
3WNqU2aWKy8Q7mblRpw_aOknW47YvZKVNQzlWdijXW5E1hQ5c6sfjqo92pq5mNKJ_Xkl71xHyp-WBfw-
xJqZ9pUuG4jnUiKgzYHvkccDF5XIMrpA67VnBozmyLckQOKEXesRD2hacrjb-
T89dcIZQHBUK1RYXGRxHCM_hPeBok",
      "kty": "RSA",
      "q": "07STzbuUh6p_iN2wzTegIQdnXBLnzPObCKt-KLSjkLtrZMv2YxM2yhMs-
56SsLR7EICFRAB2vdCzlovXKShubU9NSKgVI38qpGV9hii8rqN5N3dEM4Gscp_TR36ex1NoSC6kHBF2hsy
PfgD7U-txguZr6w9MN6rK4bAcg0TWGM0",
      "d": "IWcC4aKuPVlVGZeBhb3mla1mOsOcJuLxjpkZAu-
QaQO7phWzeGLEfln3tojKtIK6e11FEz137ow75Je0_oMAzE-
eTAMyseUTHZhn4LhmIdOwnsp9OZrMbNmLFpaF_rGYb0630xg27GdR4gC_lZvMuy1uCP1sr8Iyl1ujN7n_6
ETMYbdXPLOtEAB4Dag6XFTjy1l8aVvBxpscm8gKg64fgBGRvGY6PiUfqY3gaq_vX9SOOHVPN5WoShKA4fg
uxukRBiLLYNxDMDb3-h8pd1_fr2WayDzmcIXpxvVjRVZHt71C-
0Uhap6eRDMQocZXih8IdNV8zHUF_LeciE36fIb3oQ",
      "e": "AQAB",
      "use": "sig",
      "kid": "access",
      "qi": "3His_DaBkf_r7uDx9-8BOhOQPhcudT95XC9WyS5MrYIBtgqQi6IscHIqvtXFpjmPRey-
chO7p9msOAB_T8j_mg1l6UWOx6j4h_fyHEbOwRqfNemKng2Hs0uCrwpjgGf2eXzaBY8T9HlbFlTJAAARGh
_PePBi-F-IfAxGayj4hiM",
      "dp": "NJuYYpZAt1KUJJsdSKl6gCYPV3xrYj3iuTKYBCbYAH5jlP-
CFUIS5mnBVdnmuYKGTivsgi55DysluapwmSZ2KnoMBXXNb6dwixjvr8hSvuex1MN-
0m1udTUqHMfDW3dhGFxwJuq57VcsFAnVPl2ZfQBMAGPyRa-r7mwZo0Jmzfk",
      "alg": "RS256",
      "dq": "XL-
4IWIU6Hrh9OxrEP1VwiKkPcpqk3gGa_31_49kOXxiyH4zK6S3VECibHpEefYYFFq6B9jMLMzKYSJS2U1FU
85yZWp-GFcWL3_nRmeCgmBMMuilkIs3KeCrh58JoPoBrd4BN-rOqq_kDagQc-
uqh1a74PeKxLimucmWNExsH-E",
      "n": "z_NDmLu8M3KGvxvfJt8CAhdLdsqkskfY7vf9X9pW1LE_r31_HU85-
l6NNHeUWYbSNe6lt9YODnL8-
vTT6oCgre96byvpdYZ7Ki5KGe4fU96x0_ZF5LceUQc4l5dx6aptNi9mWgcZ9nkc2Xh83ASg9otG2YoYsAn
I1cO0TjzV9cMI7u7VON6SON9wbWFY01--ixMqxRAZuEJjbg4QAdL7DndRQXvmq1m7lv-
nnPPQ0a7ZTg7NZDEn5lMmadUlTVl5uvSNsACtC49R5kEkNCc1Hc-
3gootU5VyVPBx6IFHtNC2BiGasQAUpsDXZl7YtvBZwzYZwznUlluPiKLDk-4TtQ"
    }
  ]
}
```

The APIGWD only needs to know the Public part:

```
{
 "keys": [
 {
     "kty": "RSA",
     "e": "AQAB",
     "use": "sig",
     "kid": "access",
     "alg": "RS256",
     "n": "z_NDmLu8M3KGvxvfJt8CAhdLdsqkskfY7vf9X9pW1LE_r31_HU85-
l6NNHeUWYbSNe6lt9YODnL8-
vTT6oCgre96byvpdYZ7Ki5KGe4fU96x0_ZF5LceUQc4l5dx6aptNi9mWgcZ9nkc2Xh83ASg9otG2YoYsAn
I1cO0TjzV9cMI7u7VON6SON9wbWFY01--ixMqxRAZuEJjbg4QAdL7DndRQXvmq1m7lv-
nnPPQ0a7ZTg7NZDEn5lMmadUlTVl5uvSNsACtC49R5kEkNCc1Hc-
3gootU5VyVPBx6IFHtNC2BiGasQAUpsDXZl7YtvBZwzYZwznUlluPiKLDk-4TtQ"
   }
 ]
}
```

Now to create a token you can use https://keytool.online/, and paste the PubPrivJwks.json into the RSA Key Field and provide as Payload.

For example:

```
{
  "sub": "1234567890",
  "scope": "user operator",
  "name": "John Doe",
  "admin": true,
  "exp": 1600000000,
  "iat": 1516239022
}
```

This results in the following token:

```
eyJraWQiOiJhY2Nlc3MiLCJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdWIiOiIxMjM0NTY3ODkwI
iwic2NvcGUiOiJ1c2VyIG9wZXJhdG9yIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWUsImV4cCI
6MTYwMDAwMDAwMCwiaWF0IjoxNTE2MjM5MDIyfQ.mP0mXR96-
9gYIzh6_2saUQckKDwpC7jDFpo2m0g9YAj4DkSf4xDoxBqMRwFkntLC6NV0sxyUNzC-
nv5yBretJAbbX_hCMS5Jk392piCVMt9ucbwnCKs6xaJDJmMHI1qxyf7lCgd9nIlawme4_nQnMJ4N9RdVeI
```

```
uyv1siuNUOo9RdSE4cX2JIzlrjgoZmtcU-
nq_I7S2QTkdro2e1wPZKktTMAoG6VjGb7ieIQ5XyLKNQt9PWSZ2sHkd85MxXMRUWUcrEagW6JrV3uixeT3
QTZ3g9Y6Qb4XDPH3EXUoAHJ0V26rpqXDsB_nmNvI5CVvCUcaZLPYoSEzBUPa9NaFIBcg
```

The apigwd can decode that token and validates the token with the corresponding key in the specified JKWS file.

OIDC Authentication

If you use OpenID Connect for Authentication, that the Token is generated by the OIDC Connect server.

It is important to understand how the validation of the tokens works. Either the JWKS file which corresponds to the OIDC server is located locally on the system, or the OpenID Connect Server (issuer) is specified.

The first configuration possibility we already discussed. If the oicd connect server is specified the server provides an endpoint where the clients can download the public keys.

As an example for this configuration of an oidc-issuer here an excerpt of:

/etc/rtbrick/apigwd/config.json

```
     "oidc_issuer" : "http://<keycloak>/auth/realms/<real name>",
     "client_id" : "<client id>",
     "client_secret" : "<secret>",
     "redirect_url": "",
```

Specific information about the issuer can be found at http://<keycloak>/auth/realms/<realm name>/.well-known/openid-configuration.

If you also specify the client secret and the client id, this allows the APIGWD to redirect to the login page of the OIDC server. This is needed for browser-based applications like CTRLD UI.

15.1.3. Role Based Access Control (RBAC)

Role Based Access Control (RBAC) is an approach to restrict the system access to authorized users. The authorization model is role-based. There will be three items in a role-based modeel: sub, obj, and act.

- **sub**: the user (role) that wants to access a resource.

- **obj**: the resource that is going to be accessed
- **act**: the operation that the user performs on the resource

The RBAC Data Model is implemented in RBFS, and it allows you to define Permission or User Roles to various type of resources.

The model contains:

- **Resource Type**: The type of resource we are talking about (for example, BDS Table, BDS Object, REST)
- **Resource**: The identifier of the Resource (for example, Table Name, Rest endpoints). Regular expressions are allowed.
- **Permissions**: Indicates the action that a user is allowed to perform on the resource. The Permissions are CRUD (Create, Read Update, Delete). The permission gets a semantic with respect to the resource type.
- **Role**: The role of a user who tries to access a resource.

CTRLD Authorization Configuration

Activate or Deactivate Authorization in CTRLD

```
"auth_disabled": true
```

It is possible to specify the permissions in CTRLD exactly in the way specified above.

Where sub is the role a user needs to have, obj species the url endpoint the user wants to reach, and act is the HTTP Method the user wants to call on the endpoint.

For example:

```
{
 "permissions": [
   {"sub": "supervisor", "obj": "/*", "act": ".*" },
   {"sub": "reader", "obj": "/*", "act": "GET"},
   {"sub": ".*", "obj":
"/api/v1/rbfs/elements/\{element_name}/services/\{service_name}/proxy/*", "act":
".*"}
 ]
}
```

- The user with the role supervisor is allowed to access all rest endpoints, and act on them with all HTTP methods.
- The user with the role reader is allowed to access all rest endpoints, but can only call the HTTP GET method.
- All authenticated users are allowed to access the proxy endpoint with all HTTP methods.

To configure that policy CTRLD offers 2 endpoints:

- PUT /api/v1/ctrld/authorization/permissions
- GET /api/v1/ctrld/authorization/permissions

Please refer to API Documentation for more information.

RBFS Authorization configuration

RBFS Role Configuration via REST

```
{
 "objects": [
   { "attribute": { "role": "operator", "permission": "create|read|delete",
"resource_regex": "global.*", "resource_type": "object" } },
   { "attribute": { "role": "operator", "permission": "create|read|delete",
"resource_regex": "global.*", "resource_type": "table" } }
 ],
 "table": { "table_name": "secure.global.rbac.authorization.config", "table_type":
"authorization_config_table" }
}

{
 "objects": [
   { "attribute": { "role": "user", "permission": "-|read|-", "resource_regex":
"global.*", "resource_type": "table" } },
   { "attribute": { "role": "user", "permission": "-|read|-", "resource_regex":
"global.*", "resource_type": "object" } }
 ],
 "table": { "table_name": "secure.global.rbac.authorization.config", "table_type":
"authorization_config_table" }
}
```

- **role** : Represents role in the system
- **resource_type** : Represents resources in the RBFS (table|object).
- **resource_regex** : Regex for the resources to be accessed.
- **permission** : Bitmap representing permissions to create, read and delete.

create|read|delete

Action	BDS Table	BDS Object
Create	Create a BDS Table	Create/Update a BDS Object
Read	Read Table Header Objects or Metadata	Read BDS Objects
Delete	Delete a BDS Object	Delete a BDS Object

RBFS Authorization CLI Configurations

Global user role configuration:

set system authorization global role <name> **rbac-permission** <resource-type> <resource-regex> **permission** <permission-map>

role	Represents role in the system
resource_type	Represents resources in the RBFS (table/object).
resource_regex	Regex for the resources to be accessed.
permission	Bitmap representing permissions to create, read and delete. -/-/- -/-/delete -/read/- -/read/delete create/-/- create/-/delete create/read/- create/read/delete

Example

```
admin@rtbick: cfg> set system authorization global role admin rbac-permission
table global.* permission create/read/delete
```

Lawful user role configuration

set system authorization lawful role <name> **rbac-permission** <resource-type> <resource-regex> **permission** <permission>

role	Represents lawful interceptor (LI) role in the system
resource_type	Represents resources in the RBFS (table/object).
resource_regex	Regex for the resources to be accessed.
permission	Bitmap representing permissions to create, read and delete. -/-/- -/-/delete -/read/- -/read/delete create/-/- create/-/delete create/read/- create/read/delete

Example

```
admin@rtbick: cfg> set system authorization lawful role fbi rbac-permission table
local.* permission -/read/-
```

15.1.4. SSH with TACACS+

RBFS provides a custom pluggable authentication module that gets invoked by the stock sshd on login. The necessary configurations are pre-installed on RBFS.

RtBrick-PAM, referred to as RTB-PAM helps in landing the TACACS authentication on the appropriate user in the Ubuntu container and helps in providing necessary details for the secure management plane feature.

Once the PAM client requests TACACS for the authentication, with successful authentication TACACS responds with a few RtBrick specific details.

```
{
     rtb-deny-cmds: “clear bgp peer”
     priv_lvl : some_level
}
```

On successful authentication, the RTB-PAM module creates a token (JWT) for the logged-in ssh user.

RTB-PAM Token

Token created by the RTB-PAM module contains the same claims that are defined under the RtBrick Token section, and this token is signed with the secret_jwks.json key.

The scope claim in the rtb-token is derived from the Linux groups that the locally mapped user belongs to.

The deny commands are converted into the claim rtb-deny-cmds. Once the token is created, it is transferred to the environment variable.

```
setenv RTB_TOKEN = {
  "sub": "83692",
  "iat": 1516239022,
  "exp": 1517239022,
  "name": "Admin User",
  "preferred_username", "user1",
  "scope": "operator tacacs_priv_lvl_8"
  "rtb-deny-cmds": "^clear bgp peer"
}
```

After the RTB-PAM token is created, the CLI prompt appears. If a token is not created for the logged-in user, then the user cannot perform communication with the BD.

SSH User Prompt

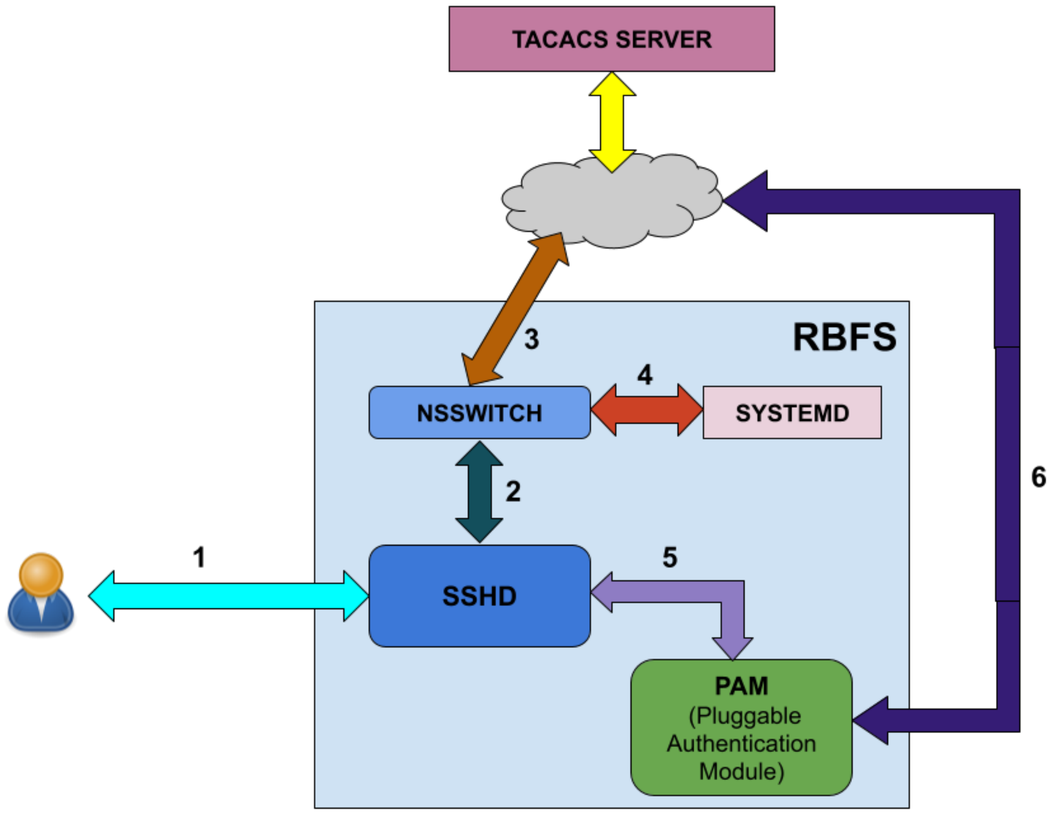
After you successfully log into RBFS via SSH, you can see the rtb-token using the shell environment. For example, an SSH prompt may look like the example below.

```
rtbng@b908f71f63b7:~$ env
SSH_CONNECTION=198.51.100.1 33136 198.51.100.44 22
LESSCLOSE=/usr/bin/lesspipe %s %s
LANG=C.UTF-8
USER=rtbng
PWD=/home/tacacs12
HOME=/home/tacacs12
SSH_CLIENT=198.51.100.1 33136 22
SUDO_USER=rtbng
PRIV_LVL=1
SSH_TTY=/dev/pts/1
SUDO_PROMPT=[sudo] password for rtbng:
MAIL=/var/mail/rtbng
TERM=xterm-256color
SHELL=/bin/bash
SHLVL=1
LOGNAME=rtbng
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/
local/games
LESSOPEN=| /usr/bin/lesspipe %s
_=/usr/bin/env
RTB_TOKEN=eyJhbGciOiJIUzI1NiIsImtpZCI6InJ0YnJpY2siLCJ0eXAiOiJKV1QifQ.eyJleHAiOjE0N
TE2MDczMDAsImlhdCI6MTQ1MTYwNjQwMCwiaXNzIjoicnRicmljay1hcGktZ3ciLCJuYW1lIjoiQmhpc2h
tYSBBY2hhcnlhIiwicHJlZmVycmVkX3VzZXJuYW1lIjoiYmhpc2htYSIsInNjb3BlIjoic3lzdGVtIiwic
3ViIjoiOTllOGI0YTEtM2E2Yi00YzI5LWJlZGItN2U3N2NjOTFjZTZiIn0.NOOAcafmHfgx-QFwiC-
_VGokbvUwrTOjhfpD9px3hMY
```

The users having access to the Linux shell only can see the installed rtb-token in the shell environment.

User Login Flow

The figure below shows the user login flow.



As shown in the figure above, the following steps are involved in the user login flow.

1. User starts to initialize using ssh
2. The SSH daemon (sshd) refers to the Name Service Switch (NSSWITCH) for user lookup
3. NSSWITCH performs a user lookup at the TACACS server

> A user lookup happens at TACACS user database even for the local users, except for the user 'supervisor'.

4. If the user is not found at TACACS user database, then the user look up happens at the local user database.
5. User authentication is performed by PAM modules
6. PAM communicates to the TACACS server for authentication if it is a TACACS user

Linux pre-configured users and groups

User Name	Group Name	Privilege
supervisor	supervisor	level 15
operator	operator	level 7-14
reader	reader	level 0-6

In-Band and Out-of-Band TACACS user SSH Login

In-band TACACS user SSH Login

A TACACS user can login using SSH to rtbrick container through inband management. RBFS should be configured with inband-management TACACS server for TACACS user login, and inband management configuration should be enabled with TACACS service.

Out-of-band TACACS user SSH Login

TACACS user can login via SSH to ONL though out-of-band management. RBFS should be configured with out-of-band management TACACS server for TACACS user login.

Configuring TACACS+ for RBFS

To configure TACACS+ server for RBFS, enter the the following commands.

Syntax

set system authorization tacacs <ipv4-address> <attribute> <value>

Command Arguments

<ipv4-address>	IPv4 address of the TACACS+ Server.
<inband | out-of-band>	Specifies inband or out-of-band connectivity to the TACACS+ server.
<inband | out-of-band> port	Specifies the TACACS+ server’s port.

<ipv4-address>	IPv4 address of the TACACS+ Server.
<inband | out-of-band> secret-encrypted-text <secret-encrypted-text>	Specifies the TACACS+ shared secret (encryption) key with a length of 4 to 16419 characters.
<inband | out-of-band> secret-plain-text <secret-plain-text>	Specifies the TACACS+ secret (encryption) key in plain text format with a . Specifies the TACACS+ global passkey in plaintext format with a length of 1 to 240 characters.

For TACACS login to work with inband management, inband configuration must be enabled with the following command:
set inband-management instance <instance-name> **tacacs true**

Example

```
root@rtbrick: cfg> set system secure-management-status true
root@rtbrick: cfg> set system authorization tacacs 192.0.2.130 inband secret-
plain-text RtBrick_Little_Secret
root@rtbrick: cfg> set inband-management instance inband_mgmt tacacs true
```

A TACACS user is not allowed to login without TACACS+ server configuration.

The example below shows the running configuration after you configure TACACS+.

```
{
  "rtbrick-config:system": {
    "authorization": {
      "tacacs": [
        {
          "ipv4-address": "192.0.2.130",
          "type": "inband",
          "secret-plain-text": "RtBrick_Little_Secret"
        }
      ]
    }
  }
}
```

Example: TACACS User Configuration in the TACACS Server

The example below shows the server configurations for rtb-deny-cmds.

```
accounting file = /var/log/tac_plus.acct
key = tacacskey

user = bob {
    login = cleartext "bob"
    member = Network_Operator
}
group = Network_Operator {
  default service = permit
  service = exec {
        priv-lvl = 10
        rtb-deny-cmds = "show bgp.*"
  }
}
```

priv-lvl is a mandatory attribute in the TACACS user configuration.

Multiple cmd-regexes can be configured with each regexes separated by semicolon (;).

Example:

```
rtb-deny-cmds = "show bgp .*;show isis .*"
```

Troubleshooting NSS User Lookup Issues

To look up the TACACS username with all NSS methods, enter the following command:

```
ubuntu@rtbrick:~$ sudo getent passwd <tac_user>
```

To look up the local user within the local user database, enter the following command:

```
ubuntu@rtbrick:~$ sudo getent -s compat passwd <local_user>
```

To look up the TACACS user within the TACACS+ server database, enter the following command:

```
ubuntu@batman:~/development$ sudo getent -s tacplus passwd <tacuser>
```

If TACACS does not appear to be working correctly, You can enable debug logging by adding the debug=1 parameter to one or more of these files:

```
/etc/tacplus_servers
/etc/tacplus_nss.con
```

Configuring Secure Management Logs

SSH User Login Logs

The transaction logs of users (in the PAM module) are available in the following log file:

```
/var/log/auth.log
```

Commands to enable secure-management logging:

set log bd <bd_name> **module secure_management logmap** <log_map> **level** <log-level>

Example:

```
root@rtbrick: cfg> set log bd all module secure_management logmap all level Info
```

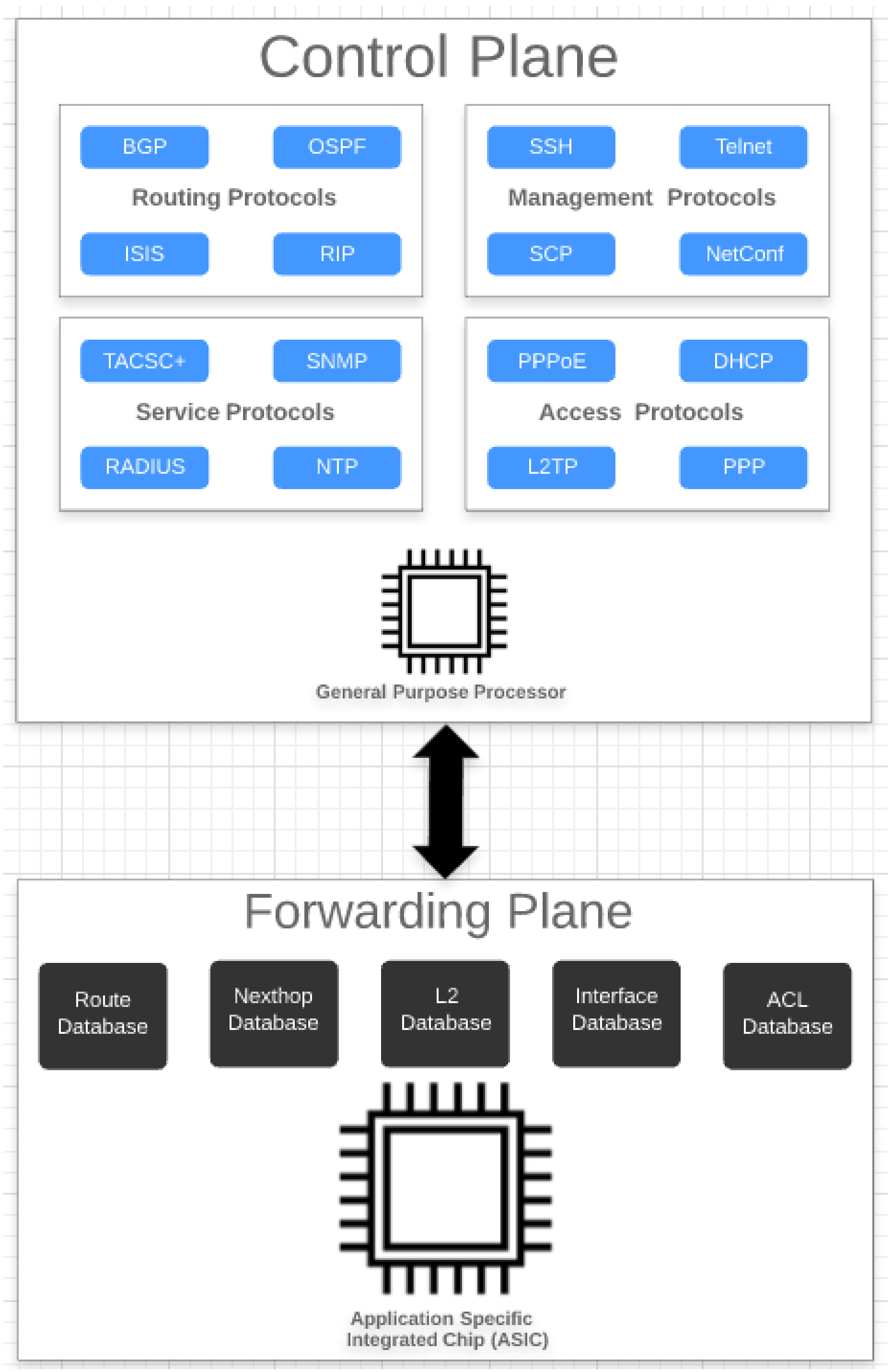
15.2. Securing Control Plane

15.2.1. Securing the Control Plane Overview

Control Plane Traffic

Control plane security enables you to filter or rate-limit unwanted traffic that is transmitted from the forwarding plane to the control plane. In RBFS, you can use Access Control Lists (ACL) and policers to secure the router’s control plane.

All routing protocols, management protocols, service protocols run in the control plane. The output of these protocols result in certain databases like routing table, MAC table, ARP table, etc., which eventually get programmed in the forwarding plane.

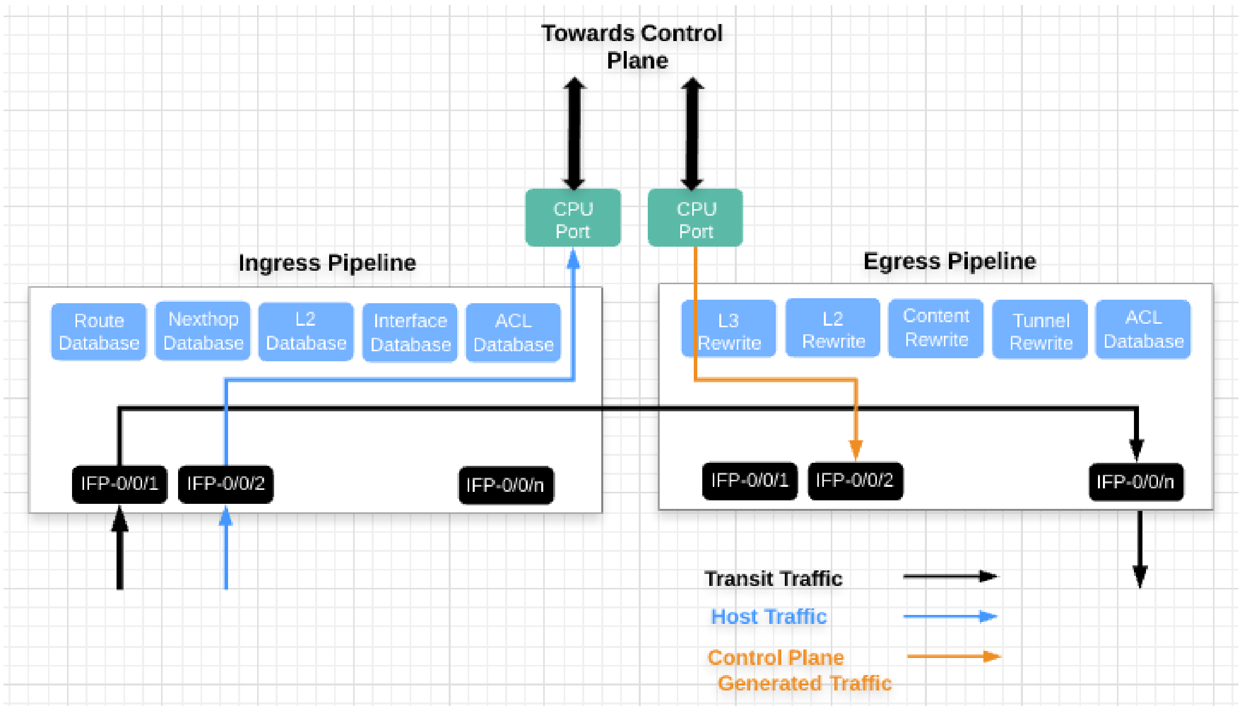


In the diagram above, you see the routing protocols (BGP, OSPF, ISIS), management protocols (SSH, RESTCONF, etc), service protocols (RADIUS, NTP, TACACS+), and access protocols (PPPoE, DHCP, L2TP, PPP) associated with the control plane. The control plane is generally implemented in software by using general-purpose processors. These protocols typically build a large number of databases like routing, switching, and ACL tables.

In contrast, a forwarding plane is associated with a copy of the databases (Routing, Switching, ACL, etc) built by the control plane. These entries typically contain the match and action which decide the packet flow in the forwarding plane. The forwarding plane functionality is realized in high performance Application Specific Integrated Circuits (ASICs) that are capable of handling very high packet rates.

There are two kinds of traffic:

1. **Control traffic**: Control traffic is destined to the device itself, that means, packets are handled by the router itself. The traffic is classified as control traffic based on matching destination IP, or because of ACL rules, or because of some kind of exception that occurred while parsing the packet (non-acceptable fields, TTL expiry, etc).
2. **Transit traffic**: Transit traffic not destined to the device itself. These packets will be sent out on one of the routers physical interfaces.



All control traffic packets will be destined to the CPU port(s). These packets are redirected to the control plane for further processing. However, general purpose processors in the control plane are not designed for packet processing, and might get overloaded if the rate of control plane traffic is too high, for example caused by a DDoS attack. Therefore you should to protect the router control plane by implementing mechanisms to filter completely or rate-limit traffic not required or unwanted at the control plane level.

Securing the Control Plane

In RBFS, there are two fundamental mechanisms how control-plane traffic is redirected to the CPU:

1. Via protocol ACLs
2. Via route lookup

Both mechanisms need to be considered and secured separately, as described in the following sections.

Control Plane Traffic via Protocol ACLs

All routing protocols (BGP, OSPF, ISIS), management protocols (SSH, RESTCONF, etc), service protocols (RADIUS, NTP, TACACS+), and access protocols (PPPoE, DHCP, L2TP, PPP), if enabled by configuration, automatically create Access Control Lists (ACLs) required to punt the protocol traffic to the control plane CPU. ACLs are the building block for securing the control plane. An ACL defines a rule, which typically contains match conditions and actions. If a packet matches the rule conditions, the associated actions will be applied. Protocol ACLs do not need to be defined by configuration. Another benefit is, they are very specific, for example match on auto-discovered IPv6 link-local neighbors. The protocol ACLs can verified using the 'show acl (detail)' command.

Example 1: Protocol ACL created by LLDP

```
supervisor@rtbrick: op> show acl detail
Rule: lldp.ifp-0/0/1.trap.rule
  ACL type: l2
  Ordinal: -
    Match:
      Attachment point: ifp-0/0/1
      Direction: ingress
      Destination MAC: 01:80:c2:00:00:0e
    Action:
      Redirect to CPU: True
    Result:
      Trap ID: LLDP
<...>
```

Example 2: Protocol ACL created by RADIUS

```
supervisor@rtbrick: op> show acl detail
Rule: radius-srv1-v4-auth-trap
  ACL type: l3v4
  Ordinal: -
    Match:
      Source L4 port: 1812
      IP protocol: UDP
    Action:
      Redirect to CPU: True
    Result:
      Trap ID: Radius
<...>
```

By default, for most of the control protocols, there is a single action Redirect to CPU: True. Thereby all traffic matching the match criteria gets punted to the CPU without any rate limit. There is one exception, for PPPoE only the traffic is rate-limited by default. As shown in the following example, there is an additional action Policer profile name: created by default that limits the PPPoE traffic to 50 Mbps per session:

Example 3: Protocol ACL with Policer created by PPPoE

```
supervisor@rtbrick: op> show acl detail
Rule: pppoed_hostif-0/0/1_7-7-1-4090_8863
  ACL type: PPPOE
  Ordinal: -
    Match:
      Attachment point: ifl-0/0/1
      Ethertype: 34915
    Action:
      Redirect to CPU: True
      Policer profile name: _DEFAULT_POLICER_50_MB
    Result:
      Trap ID: PPPoE
<...>
```

For all other protocols, rate limiting needs to enabled by configuration in order to secure the control plane. This is described in section 2.1 below.

Control Plane Traffic via Route Lookup

By default, any other traffic destined to one of the router’s IP addresses, commonly referred to as "my IP", and not matching any ACL is redirected to the CPU via a route lookup. This applies to loopback as well physical interface addresses. In order to secure the control plane against malicious traffic sent to one of the router’s IP addresses (not matching any protocol ACL), ACLs need to be defined by configuration. In RFBS, such "manually" created ACLs are referred to user-defined ACLs.

Typically you will want to completely block some unwanted traffic sent to "my IP", but allow and rate-limit some required traffic like for example ICMP. Please note, when designing the security ACLs to protect "my IP", you do NOT need to consider the protocol traffic already handled by the protocol ACLs.

When configuring ACLs, protocol ACLs and user-defined ACLs may conflict. For example, an ACLs created by the BGP routing protocol might match on TCP traffic sent to the routers loopback address with port 179. A user-defined ACL however might deny any traffic sent to this loopback address. In this case, protocol ACLs shall take precedence over user-defined ACLs, so that you do not accidentally break the protocol operation. In RBFS, this is implemented using different ACL database priorities.

Configuring ACLs to protect "my IP" is described in section 2.2 below.

Limitations and Notes

- The control-plane security features are supported on hardware platforms. They are not supported on virtual deployments.
- On the Edgecore AS5916-54XKS platform, BGPv6 with link-local peering uses route lookup instead of protocol ACLs. Therefore traffic sent to IPv6 link local addresses cannot be restricted via ACL to not break BGPv6 link-local peerings.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Control Plane security is currently not supported for PIM, IGMP, and L2TP protocol traffic.

15.2.2. Control Plane Security Configuration

As highlighted above, there are two fundamental mechanisms how traffic is redirected to the CPU, via protocol ACLs, and via route lookup. These are addressed in following two sections.

Secure Control Plane Traffic via Protocol ACLs

This section describes the configuration options for control plane traffic that is redirected to the CPU via protocol ACLs. These ACLs are automatically created by the protocols, and do not need to be - and cannot be - configured manually. For example if you configure a routing protocol like BGP, the required ACLs to match and punt the BGP packets to the control plane are created automatically.

Enabling the Control Plane Security Feature

By default, all packets matching the protocol ACLs will be sent to the control plane without any rate limit, except for PPPoE. The RBFS Control Plane Security feature allows to add policers to all protocol ACLs. If enabled, this feature creates a set of default policers, and applies them to the protocol ACLs. Thereby the control plane gets secured against DDoS attacks matching these ACLs.

Syntax:

set forwarding-options control-plane-security <attribute> <value>

Attribute	Description
state (enable|disable)	Enable or disable the control-plane security feature. Default: enable.

Example:

```
{
    "rtbrick-config:forwarding-options": {
      "control-plane-security": {
        "state": "enable"
      }
    }
}
```

Configuring Host Path QoS

The host-path-qos enable feature is disabled by default. Once it is enabled, you cannot disable it.

To enable the `host-path-qos' feature, enter the following command:

Syntax:

set forwarding-options class-of-service control-plane-qos ingress-qos <attribute> <value>

Attribute	Description
state (enable|disable)	Enable or disable the host path QoS feature. Default: disabled.

Example

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options class-of-service
control-plane-qos
{
    "rtbrick-config:control-plane-qos": {

      "ingress-qos": {
        "state": "enable"
      }
    }
  }
```

By enabling this feature, the default scheduler and queue configurations are installed and the CPU ports queue mapping will change. Also, all control plane ACLs will be reprogrammed to update action_forward_class.

Marking Outbound Control Plane Traffic

RBFS enables you to configure the various protocols to mark the egress control plane traffic. The control plane traffic can be marked with type-of-service (ToS) values.

- For BGP, OSPF, RADIUS, PIM, L2TPv2, LDP, and DHCP protocols, the remark-type should be configured as ToS
- For the IGMP and PPPoE protocols, the remark-type can be be configured as p-bit or tos
- The outbound-marking attributes such as name, code-point and remark-type are mandatory
- If the name of the protocol is L2-all, then the remark-type should configured as p-bit
- If the name of the protocol is L3-all, then the remark-type should configured as tos
- Redundancy related control traffic can be marked with specific tos value with the L3-all option

set forwarding-options class-of-service control-plane-qos outbound-marking protocol <protocol-name> <remark-type-value> **codepoint** <codepoint-value>

Option	Description
<protocol-name>	Specifies the protocol name.
<remark-type-value>	Specifies the remark type value that can be p-bit or tos.
<codepoint-value>	Specifies the codepoint value. The supported range for p-bit outbound-marking is 0-7.

Example: Marking Outbound CP Traffic Configuration

```
supervisor@rtbrick>LEAF01: cfg> show config
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "class-of-service": {
        "control-plane-qos": {
          "outbound-marking": {
            "protocol": [
              {
                "protocol": "bgp",
                "remark-type": "tos",
                "codepoint": 192
              },
              {
                "protocol": "dhcp",
                "remark-type": "p-bit",
                "codepoint": 5
              },
              {
                "protocol": "l3-all",
                "remark-type": "tos",
                "codepoint": 224
              },
                    {
                "protocol": "igmp",
                "remark-type": "p-bit",
                "codepoint": 7
              },
              {
                "protocol": "igmp",
                "remark-type": "tos",
                "codepoint": 192
              },
              {
                "protocol": "ppp",
                "remark-type": "p-bit",
                "codepoint": 7
              },
              {
                "protocol": "ppp",
                "remark-type": "tos",
                "codepoint": 192
              },
              {
                "protocol": "radius",
                "remark-type": "tos",
                "codepoint": 100
            }
            ]
          }
        }
      }
    }
  }
}
```

Marking of Outbound Control Plane VLAN P-bits for ARP/ND

You can configure VLAN P-bit marking for ARP and ND packets in outbound control plane traffic.

When a user configures the p-bit for both the ARP protocol and "l2-all," the setting for the ARP protocol takes priority. If the p-bit for the ARP is removed, the system will then apply the "l2-all" configuration. This same behavior applies to the ND protocol as well.

Syntax:

set forwarding-options class-of-service control-plane-qos outbound-marking protocol <protocol> **p-bit codepoint** <codepoint>

Example: Configuring p-bit for ARP/ND

```
set forwarding-options class-of-service control-plane-qos outbound-marking
protocol arp p-bit codepoint 5
set forwarding-options class-of-service control-plane-qos outbound-marking
protocol nd p-bit codepoint 4
```

Example:

```
"rtbrick-config:forwarding-options": {
    "class-of-service": {
        "control-plane-qos": {
        "outbound-marking": {
            "protocol": [
            {
                "protocol": "arp",
                "remark-type": "p-bit",
                "codepoint": 5
            },
            {
                "protocol": "nd",
                "remark-type": "p-bit",
                "codepoint": 4
            }
            ]
        }
        }
    }
    }
```

Configuring Control Plane QoS Per IFP

The marking of control plane packets can be configured on a per-IFP basis for PPPoE, whereas it was previously set globally for each protocol in earlier releases of RBFS.

The priority of p-bit/ToS settings is processed in this order: IFP (highest), protocol level, and l2-all/l3-all (lowest).

The table below shows each protocol’s supported QoS types and IFP support.

Protocol	QoS Type Supported	Support for IFP-based QoS
BGP	tos	No
OSPF	tos	No
LDP	tos	No
ISIS	p-bit	No
IPoE	tos	No
DHCP Relay	p-bit	No
RADIUS	tos	No
PPPoE	p-bit/tos	Yes (only p-bit)
IGMP	p-bit/tos	No
L2TP	tos	No

Protocol	QoS Type Supported	Support for IFP-based QoS
ARP/ND	p-bit	No
PIM	tos	No

Syntax:

set forwarding-options class-of-service control-plane-qos outbound-marking protocol <protocol-name> [p-bit / tos] **ifp** <ifp-name> **codepoint** <codepoint>

Configuration Example:

```
supervisor@rtbrick>: cfg> set forwarding-options class-of-service control-plane-
qos outbound-marking protocol ppp p-bit ifp ifp-0/0/1 codepoint 3
supervisor@rtbrick>: cfg> set forwarding-options class-of-service control-plane-
qos outbound-marking protocol ppp p-bit ifp ifp-0/0/2 codepoint 4
supervisor@rtbrick>: cfg> commit
```

The example below shows the result of the configuration.

```
"rtbrick-config:forwarding-options": {
  "class-of-service": {
    "control-plane-qos": {
      "outbound-marking": {
        "protocol": [
          {
            "protocol": "ppp",
            "remark-type": "p-bit",
            "ifp": [
              {
                "ifp": "ifp-0/0/1",
                "codepoint": 3
              },
              {
                "ifp": "ifp-0/0/2",
                "codepoint": 4
              }
            ]
          }
        ]
      }
    }
  }
}
```

Viewing Control Plane QoS Details

The command shows qos control-plane displays QoS remarking types, interface names, and codepoints for all or specified protocols.

The following example displays QoS control plane information for the protocol 'ppp'.

```
supervisor@rtbrick>: cfg> show qos control-plane protocol ppp
Protocol         Remark Type       Interface         Codepoint
ppp              p-bit                   all              5
                                 ifp-0/0/1              3
                                 ifp-0/0/2              4
```

Restricting Management Access

If you enable inband management access for example via SSH, protocol ACLs will be created that match on the enabled protocols and redirect the management traffic to the control plane. By default, this traffic is not restricted in terms of source IP addresses. You can optionally restrict management access to trusted IP addresses by applying a source prefix list. An additional match condition will then be added to the protocol ACLs for inband management.

Configuring a Prefix List

Syntax:

set forwarding-options prefix-list <options>

Option	Description
<prefix-list-name> ipv4-prefix <ipv4_prefix>	Prefix list configuration for IPv4.
<prefix-list-name> ipv6-prefix <ipv6_prefix>	Prefix list configuration for IPv6.

Applying a Prefix List

Syntax:

set inband-management instance <instance-name> source-prefix-list <list-name>

Example: Inband Management Configuration with Source Prefix List

```
"rtbrick-config:inband-management": {
      "instance": [
        {
          "name": "default",
          "ssh": "true",
          "ntp": "true",
          "source-prefix-list": "list1"
        }
      ]
    },

"rtbrick-config:forwarding-options": {
      "prefix-list": [
        {
          "prefix-list-name": "list1",
          "ipv4-prefix": [
            {
              "ipv4-prefix": "198.51.100.100/24"
            },
            {
              "ipv4-prefix": "198.51.100.33/24"
            },
            {
              "ipv4-prefix": "198.51.100.44/24"
            }
          ]
        }
```

The inband management is provided to only the source address specified in the prefix list if the prefix list is configured. If the prefix list not configured, it works for all source IPs. Also, the prefix addresses configured should be of /32.

Configuring Protocol ACL Options

This section describes how to configure policers per protocol and configure match on IPv4 ToS or IPv6 TC fields for protocol ACLs.

- If control plane security is disabled, this configuration has no effect in the system
- Protocol-specific configuration will take priority over ALL configuration in the control-plane-security protocol

Syntax:

set forwarding-options control-plane-security protocol <protocol-name> <attribute> <value>

Option	Description
protocol <protocol-name>	Name of the protocol. You can configure individual protocols, and/or all protocols using the 'ALL' value keyword.
match-tc <tc-value>	Configure IPv6 TC value. The range is 0 to 248.
match-tos <tos-value>	Configure IPv4 ToS value. The range is 0 to 248.
policer <policer>	Configure policer name.

Example:

```
{
  "ietf-restconf:data": {
    "rtbrick-config:forwarding-options": {
      "class-of-service": {
        "policer": [
          {
            "policer-name": "_DEFAULT_POLICER_BGP_LL",
            "flags": "color-blind",
            "level1-rates": {
              "cir": 1000,
              "cbs": 1000,
              "pir": 1200,
              "pbs": 1000
            },
            "levels": 1,
            "type": "two-rate-three-color"
          }
        ]
      },
      "control-plane-security": {
        "state": "enable",
        "protocol": [
          {
            "protocol": "PIM",
            "policer": "_DEFAULT_POLICER_PIM",
            "match-tos": 192,
            "match-tc": 120
          }
        ]
      }
    }
  }
}
```

Secure Control Plane Traffic via Route Lookup

This section describes the configuration to secure the control plane for traffic that is redirected to the CPU via route lookup. Any packet sent to one of the router’s IP addresses ("my IP") and not matching any ACL, will be redirected to the CPU via route lookup. By default this type of traffic is not restricted or rate-limited. In order to secure the control plane, you need to apply ACLs by configuration. Please note you do not need to consider and allow any protocol traffic that is already captured by the automatically created protocol ACLs. You only need to explicitly define rules for any other traffic sent to "my IP". In the simplest case, you can deny any other traffic sent to the router. Typically you will want to allow some additional traffic like ICMP, and deny anything else.

Configuring ACLs

This section describes how to configure ACLs to secure the control plane to protect "my IP". In RFBS, ACLs are applied globally, that is, you do not need to attach them by configuration. Besides, for ACLs matching traffic sent to one of the router's IP addresses, the redirect-to-cpu action applies implicitly and does not need to be configured.

Syntax:

set forwarding-options acl <options>

Option	Description
l3v4	ACL configuration for IPv4
l3v6	ACL configuration for IPv6
rule <rule-name>	Name of the ACL rule
ordinal <ordinal-value>	Number of the configuration entry. Please note the order of the configuration entries (ordinals) does not determine the processing.
match <condition>	Supported match conditions are IP source/destination prefix, prefix lists, source/destination Port, IP protocol, and direction.
action <action>	Supported actions are permit, drop, and police.
priority <value>	ACL entry priority. Determines the processing precedence for multiple matching i.e. conflicting rules. A less-specific rule should have a lower priority so that a more-specific rules takes precedence. Default: 10.

Example 1: Denying any Traffic destined to the Router's Loopback Addresses

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options acl
{
    "rtbrick-config:acl": {
      "l3v4": {
        "rule": [
          {
            "rule-name": "BLOCK_LOOPBACK_TRAFFIC_v4",
            "ordinal": [
              {
                "ordinal-value": 20,
                "match": {
                  "destination-ipv4-prefix": "198.51.100.43/24",
                  "direction": "ingress"
                },
                "action": {
                  "drop": "true"
                },
                "priority": 20
              }
            ]
          }
        ]
      },
      "l3v6": {
        "rule": [
          {
            "rule-name": "BLOCK_LOOPBACK_TRAFFIC_v6",
            "ordinal": [
              {
                "ordinal-value": 20,
                "match": {
                  "destination-ipv6-prefix": "2001:db8:0:10::/32",
                  "direction": "ingress"
                },
                "action": {
                  "drop": "true"
                },
                "priority": 20
              }
            ]
          }
        ]
      }
    }
  }
supervisor@rtbrick>LEAF01: cfg>
```

Example 2: ACL allowing and rate-limiting ICMPv4/v6, and denying any other Traffic

```
supervisor@rtbrick>LEAF01: cfg> show config forwarding-options acl
{
    "rtbrick-config:acl": {
      "l3v4": {
        "rule": [
          {
            "rule-name": "BLOCK_LOOPBACK_TRAFFIC_v4",
            "ordinal": [
              {
                "ordinal-value": 20,
                "match": {
                  "destination-ipv4-prefix": "198.51.100.43/24",
                  "direction": "ingress",
                  "ip-protocol": "ICMP"
                },
                "action": {
                  "permit": "true"
                },
                "priority": 20
              }
            ]
          }
        ]
      },
      "l3v6": {
        "rule": [
          {
            "rule-name": "BLOCK_LOOPBACK_TRAFFIC_v6",
            "ordinal": [
              {
                "ordinal-value": 20,
                "match": {
                  "destination-ipv6-prefix": "2001:db8:0:10::/32",
                  "direction": "ingress",
                  "ip-protocol": "IPv6_ICMP"
                },
                "action": {
                  "permit": "true"
                },
                "priority": 20
              }
            ]
          }
        ]
      }
    }
  }
```

15.2.3. Control Plane Security Operational Commands

Show Commands

This section describes operational commands available to verify various control-plane security features.

Verifying ACLs

The show acl command allows to verify protocol ACLs as well as user-defined ACLs.

Syntax:

show acl <options>

Option	Description
detail	Displays all ACL details
<acl-name>	Displays the details for a single ACL

Example 1: Protocol ACL with Control-Plane Security enabled

```
supervisor@rtbrick>LEAF01: op>  show acl detail

Rule: lldp.ifp-0/0/1.trap.rule
  ACL type: l2
  Ordinal: -
    Match:
      Attachment point: ifp-0/0/1
      Direction: ingress
      Destination MAC: 01:80:c2:00:00:0e
    Action:
      Redirect to CPU: True
      Policer profile name: _DEFAULT_POLICER_50_MB
    Result:
      Trap ID: LLDP
<...>
Rule: radius-srv1-v4-auth-trap
  ACL type: l3v4
  Ordinal: -
    Match:
      Source L4 port: 1812
      IP protocol: UDP
    Action:
      Redirect to CPU: True
      Policer profile name: _DEFAULT_POLICER_20_MB
    Result:
      Trap ID: Radius
<...>
```

Example 2: ACL for Inband Management with Source Prefix List

```
supervisor@rtbrick>LEAF01: op>  show acl detail

Rule: ifm.inband.mgmt.lo-0/0/0/1.ssh.client.v4.trap.rule.1
  ACL type: l3v4
  Ordinal: 1
    Match:
      Destination IPv4 address: 198.51.100.91
      Source IPv4 address: 198.51.100.92
      Source L4 port: 22
      IP protocol: TCP
    Action:
      Redirect to CPU: True
    Result:
      Trap ID: INBAND
```

Example 3: User-defined ACL to Protect "my IP"

```
supervisor@rtbrick>LEAF01: op> show acl Protect-CP-v4

Rule: Protect-CP-v4
  ACL type: l3v4
  Ordinal: 1
    Match:
      Direction: ingress
      Destination IPv4 prefix: 198.51.100.91/24
      Source IPv4 prefix: 198.51.100.90/24
      IP protocol: ICMP
    Action:
      Permit: True
    Result:
      Trap ID: User Defined
  Ordinal: 2
    Match:
      Direction: ingress
      Destination IPv4 prefix: 198.51.100.91/24
    Action:
      Drop: True
    Priority: 5
    Result:
      Trap ID: User Defined
```

Verifying ACL Counters

The "show acl statistics" command displays information about the ACL packet counters. The counters are useful to verify if the ACL rules actually match, and if potentially malicious traffic gets dropped.

Syntax:

show acl statistics

Example 1: ACL statistics information

```
supervisor@rtbrick>LEAF01: cfg> show acl statistics
ACL                                                                        Units      Total
Accepted    Dropped
lldp.ifp-0/0/12.trap.rule                                                  Packets    -          -
-
                                                                           Bytes      -          -
-
lldp.ifp-0/0/16.trap.rule                                                  Packets    -          -
-
                                                                           Bytes      -          -
-
lldp.ifp-0/0/27.trap.rule                                                  Packets    -          -
-
                                                                           Bytes      -          -
-
lldp.ifp-0/0/53.trap.rule                                                  Packets    -          -
-
                                                                           Bytes      -          -
-
default_bgp_l4_trap_12::2_12::1_dst                                        Packets    12         12
0
                                                                           Bytes      1353       1353
```

```
0
default_bgp_l4_trap_12::2_12::1_src                                       Packets    12         12
0
                                                                          Bytes      1353       1353
0
default_bgp_l4_trap_12.0.0.2_12.0.0.1_dst                                 Packets    12         12
0
                                                                          Bytes      1353       1353
0
default_bgp_l4_trap_12.0.0.2_12.0.0.1_dst                                 Packets    -          -
-
                                                                          Bytes      -          -
-
default_bgp_l4_trap_12.0.0.2_12.0.0.1_src                                 Packets    12         12
0
                                                                          Bytes      1353       1353
0
default_bgp_l4_trap_12.0.0.2_12.0.0.1_src                                 Packets    -          -
-
                                                                          Bytes      -          -
-
supervisor@rtbrick: cfg>
```

Example 2: Display ACL statistics information for the specified ACL

```
supervisor@rtbrick>LEAF01: cfg> show acl default_bgp_l4_trap_12.0.0.2_12.0.0.1_dst statistics
ACL                                                                       Units      Total
Accepted    Dropped
default_bgp_l4_trap_12.0.0.2_12.0.0.1_dst                                 Packets    20         20
0
                                                                          Bytes      1917       1917
0
default_bgp_l4_trap_12.0.0.2_12.0.0.1_dst                                 Packets    -          -
-
                                                                          Bytes      -          -
-
supervisor@rtbrick>LEAF01: cfg>
```

Verifying Control Plane Policers

This command allows to view the policers created by the control-plane security feature.

Syntax:

show qos policer <options>

Option	Description
-	Displays all policers created by the control-plane security feature
<policer-name>	Displays information about the specified policer
counter	Displays all policer counters

Example 1: Display information of all policers created by the control-plane security

feature

```
supervisor@rtbrick>LEAF01: cfg> show qos policer
Policer: _DEFAULT_POLICER_100_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        100000         100000         33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: _DEFAULT_POLICER_1_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        1000           1000           33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: _DEFAULT_POLICER_20_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        20000          20000          33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: _DEFAULT_POLICER_250_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        250000         250000         33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: _DEFAULT_POLICER_500_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        500000         500000         33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: _DEFAULT_POLICER_50_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        50000          50000          33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
Policer: _DEFAULT_POLICER_5_MB
Active: True, Type: two-rate-three-color, Levels: 1, Flags: -
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        5000           5000           33000          33000          -              -
  2        -              -              -              -              -              -
  3        -              -              -              -              -              -
  4        -              -              -              -              -              -
supervisor@rtbrick>LEAF01: cfg>
```

Example 2: Display information of a specific policer

```
supervisor@rtbrick>LEAF01: cfg> show qos policer Premium_Upstream_Hierarchical_Policer
Policer: Premium_Upstream_Hierarchical_Policer
Active: False, Type: two-rate-three-color, Levels: 4, Flags: color-blind
  Level    CIR(Kbps)      PIR(Kbps)      CBS(KB)        PBS(KB)        Max CIR(Kbps)  Max PIR(Kbps)
  1        1000           1200           1000           1000           -              -
  2        900            1000           1000           1000           -              -
  3        5000           5200           1000           1000           -              -
  4        6000           6200           1000           1000           -              -
```

Example 3: Display information of policer counter

```
supervisor@rtbrick>LEAF01: cfg> show qos policer counter
Interface                         Level  Units     Total            Received         Dropped
ipv6_ll_prefix_acl                1      Packets   48               48               0
                                         Bytes     6383             6383             0
ipv6_mcast_ff01_prefix_acl        1      Packets   48               48               0
                                         Bytes     6383             6383             0
ipv6_mcast_ff02_prefix_acl        1      Packets   48               48               0
                                         Bytes     6383             6383             0
ppp-0/1/28/72339069014638594      1      Packets   0                0                0
                                         Bytes     0                0                0
ppp-0/1/28/72339069014638594      2      Packets   0                0                0
                                         Bytes     0                0                0
ppp-0/1/28/72339069014638594      3      Packets   0                0                0
                                         Bytes     0                0                0
ppp-0/1/28/72339069014638594      4      Packets   0                0                0
                                         Bytes     0                0                0
pppoed_ifp-0/1/28_1-3500-1-35     1      Packets   48               48               0
                                         Bytes     6383             6383             0
pppoed_ifp-0/1/28_1-3500-1-35     1      Packets   48               48               0
                                         Bytes     6383             6383             0
pppoed_ifp-0/1/30_1-3500-1-35     1      Packets   48               48               0
                                         Bytes     6383             6383             0
pppoed_ifp-0/1/30_1-3500-1-35     1      Packets   48               48               0
                                         Bytes     6383             6383             0
```

The show qos policer counter command displays the policer-level counters for the subscribers. The packets that get dropped after the RPF check, are currently updated in the local.bcm.q2c.trap.stats table in FIBD.

15.3. Local User Management

15.3.1. Local User Management Overview

Local User Management enables you to create, manage, and secure the Linux local users and groups through the RBFS configuration. This enables you to manage users and groups in the following environments:

- In the RBFS container only for the virtual platform
- In the RBFS container and on the ONL host for the hardware platforms

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

15.3.2. Local user management Configuration

RBFS allows you to create privileges that are configurable for user-defined and pre-defined roles. RBFS supports a combination of permit and deny regular expressions and a configurable default privilege to support both blacklisting and whitelisting of users. If both permit and deny command regular expressions match, the allow regular expression takes precedence.

Creating Roles

To create a role, you need to configure the following:

- Configure role-based access control (RBAC) privilege for the role
- Configure the command privilege for the role

It is important to have secure management enabled when granting any type of privilege, whether it is RBAC or command-based, to any user. Without secure management, privileges do not work.

For information about enabling Secure Management, see the section "Configuring Secure Management Logs" of the *Securing Management Plane* user guide.

Configuring the RBAC Privilege

You need to configure the RBAC privilege for both table and object.

```
set system authorization global role <name> rbac-permission ( object | table ) <resource> <permission-type>
```

Command arguments

<name>	Authorization role name
<resource>	Represents resources in the RBFS (table/object)

<permission-type>	Permissions to create, read and delete. The following are the supported RBAC permission types: -/-/- -/-/delete -/read/- -/read/delete create/-/- create/-/delete create/read/- create/read/delete

Configuring the Command Privilege

set system authorization global role <name> **cmd-permission** (**allow-cmds** <allow-cmds> | **deny-cmds** <deny-cmds>)

<role>	Authorization role name
<allow-cmds>	List of allow commands regular expression
<deny-cmds>	List of deny commands regular expression

- If you configure a privilege for any of the pre-defined roles (supervisor, operator, reader), then it replaces the default privilege.
- If you delete the configured privilege for a pre-defined role, then it will revert to the default privilege of the role.
- Priority of privilege rules is as follows: explicit deny, explicit permit, default privilege.

The example below shows the new role named "support" which has RBAC permission to read any table and objects. Also, the user is denied everything except the allowed commands (ping, set, show, traceroute, and watch-mode).

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "authorization": {
        "global": {
          "role": [
            {
              "name": "support",
              "rbac-permission": [
                {
                  "permission": "-/read/-",
                  "resource-type": "object",
                  "resource": ".*"
                },
                {
                  "permission": "-/read/-",
                  "resource-type": "table",
                  "resource": ".*"
                }
              ],
              "cmd-permission": {
                "allow-cmds": [
                  "ping .*",
                  "set .*",
                  "show .*",
                  "traceroute .*",
                  "watch .*"
                  ],
                "deny-cmds": ".*"
              }
            }
          ]
        }
      }
    }
  }
}
```

Linux pre-configured users, roles and privileges

User Name	Role Name	Default Privileges
supervisor	supervisor	Allow all actions.
operator	operator	Allow all actions except for the "reboot" action.

reader	reader	All commands will be denied other than the commands which match any of the below regular expressions.
		`"color.*",`
		`"date.*",`
		`"exit.*",`
		`"history.*",`
		`"paging.*",`
		`"ping.*",`
		`"show.*",`
		`"traceroute.*",`
		`"watch-mode.*"`

Creating New Users

The new users created through local user management will always have a primary group with the same name and ID of the created user. The new user’s ID will be allocated within the range of 3000 and 3999.

> You cannot use usernames such as root, wheel, admin, sudo or any of the SMP Linux pre-configured users and groups such as supervisor, operator, reader. Also, a username cannot start with “rtbrick_”. If a Linux user with the same username already exists but has an ID outside of the 3000-3999 range then the user creation through the RBFS configuration will fail.

To create a new user, enter the following command:

```
set system user <username>
```

Command arguments

<username>	Name of the local user

Assigning Roles to Users

A "role" is an RBFS RBAC construct and it is mapped to a Linux group. The list of user roles from the RBFS configuration becomes the list of additional Linux groups that the Linux user belongs to. You can create new users and assign “roles” to the new users. The supervisor, operator, and reader are the pre-defined and pre-configured roles both in Linux and RBFS.

When a user is configured in RBFS under “system users”, RBFS/confd validates that the list of user roles only contain roles that are pre-defined or that are configured under “system authorization”.

> Do not create role names that start with “rtbrick_”. In addition, "root", "wheel", "admin", and "sudo" are not acceptable role names.

To assign a role to a new user, enter the following command:

set system user <username> **role** <role>

Command arguments

<username>	Name of the user
<role>	Role of the user (not the primary role)

Example: Assigning Roles to Users

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "user": [
        {
          "username": "bob",
          "role": [
            "operator"
            ],
          "shell": "/usr/local/bin/cli",
          "password-hashed-text":
"$6$uTE4OYn0iRq.Vppe$JBVMQ5DZHfuCuUP5yTnfl9IJsRLQAXqTLlLMKRO8bCz9WDlB2ele8puwMrT4/
QDF2nNOcoHtqYqFljly4B.Vu0"
        }
      ]
    }
  }
}
```

Configuring Authentication for a New User

There are two ways in which you can create a password for a new user in RBFS:

1. Creating hashed password
2. Creating plain-text password

Creating Hashed Password

You can verify the integrity of your password using hashed passwords. When a user is present in the configuration but a "password hashed text" is not present, the password authentication is considered disabled for that specific user.

Three predefined roles include supervisor, operator, and reader. supervisor is the default password for the role supervisor. For the 'operator' and 'reader' roles, there is no default password.

Use the system users supervisor command to disable the default supervisor password.

You can also disable password authentication for any of the predefined users such as supervisor, operator and reader by adding a "system users supervisor" configuration section without any "password hashed text" and thus disabling password authentication for the supervisor user.

SSH public keys can be configured even if "password hashed text" is not present.

To create a password hashed text and authenticate the new user, perform the following steps:

1. Generate hash password on any Linux server.
   mkpasswd --method=SHA-512
2. Configure authentication using a password hashed text and an SSH public key.
   set system user <username> **password-hashed-text** <password-hashed-text>
   set system user <username> **ssh-pub-key** <ssh-pub-key>

Changing an Existing Hashed Password for Supervisor

You can change an existing hashed password for a 'supervisor' user after logging in to the system.

Syntax:

set system user <username> **password-hashed-text** <password-hashed-text>
set system user <username> **ssh-pub-key** <ssh-pub-key>

Example Command:

```
set system user supervisor password-hashed-text
$6$HIPXmd0uiLD0RtcT$3V7YxFJWR3b6NEGGd41RCT0TseWgdKLFAl6RecvDOqIUnaCHMt0zo0ZZR4/1
```

Command arguments

<username>	Name of the user.
<password-hashed-text>	Password string.

<ssh-pub-key>	public keys of a user. You can specify multiple ssh-pub-keys.

3. Log in using username and password hashed text.

- It is possible to change shell, password-hash and ssh-pub-keys for supervisor, reader, and operator roles.
- The password string provided as part of the RBFS configuration needs to be a compatible password hashed text as defined by the shadow manual page: https://manpages.debian.org/buster/passwd/shadow.5.en.html and by the crypt https://manpages.debian.org/buster/manpages-dev/crypt.3.en.html.

Example

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "user": [
        {
          "username": "bob",
          "shell": "/usr/local/bin/cli",
          "password-hashed-text":
"$5$L2DaOYYuddhBV$9RA5MX9RQzLC9fIKJzbnoFBb88w9rkSXl7GVrVJ9PY7",
          "ssh-pub-key": [
            "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQCBAAABAQCubg5sdDycPN5EViNkV6w7rfp2GAfKWuInfaL3xOXyvSNpsmaHIL
YmgrLUU0GKQH9gauPUJpDcvvYaMt0ZBuTbWHVMUc4cvhgbNDkTB2bG2cTZ5QzbicyXff3BlDWQThVp2LtV
BiW2tf7JTTa9SnL4Lnm+CQcXsQ0rxqy2S6bJpsRYlFMyQl/hZ4QEWE153dw0HGvcG8mjfnPN4wvCc/omfD
3ljxx+Gf4oFS0davX6pdphUKLvgL33VVG5xaK71imv2l3897LIJZaHxy7FbB+CjSYT6QNq1XksX8omrbRj
iP3enEQi/bANtzTNnGDnIm1KHf3xuKpoKw+B5fhDZogx"
            ]
        }
      ]
    }
  }
}
```

Creating Plain-Text Password

To configure a password with plain text password for a new user, use the following command.

Syntax:

set system user <username> **password-plain-text** <password-plain-text>

Command arguments

<username>	Name of the user
<password-plain-text>	Specifies the plain-text password

Example for Configuring Plain Text Password

```
set system user bob password-plain-text bob123
set system user bob shell /usr/local/bin/cli
set system user bob role operator
```

Changing an Existing Plain-Text Password for Supervisor

You can change an existing plain text password for the 'supervisor' user after logging into the system. Use the following commands to change an existing password.

Syntax:

set system user <username> **password-plain-text** <password-plain-text>

Sample Commands:

```
set system user supervisor shell /bin/bash
set system user supervisor password-plain-text rtbrick
```

Viewing Configuration of Plain Text Password

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "user": [
        {
          "username": "bob",
          "role": [
            "operator"
            ],
          "shell": "/usr/local/bin/cli",
          "password-hashed-text":
"$6$uTE4OYn0iRq.Vppe$JBVMQ5DZHfuCuUP5yTnfl9IJsRLQAXqTLlLMKRO8bCz9WDlB2ele8puwMrT4/
QDF2nNOcoHtqYqFljly4B.Vu0"
        }
      ]
    }
  }
}
```

Setting the User Shell

RBFS validates that the shell is one of the following 3 valid options:

- /usr/sbin/nologin
- /bin/bash
- /usr/local/bin/cli

To configure user shell, enter the following command:

set system user <username> **shell** <shell>

Command arguments

<username>	Name of the user
<shell>	Name of the shell

Example

```
root@rtbrick: cfg> set system user smith shell /usr/local/bin/cli
```

Specifying the Display Name for User Names

The display name allows you to specify a preferred name so that you can easily identify the user. You can change your display name by entering the following command:

set system user <username> **display-name** <display_name>

Command arguments

<username>	Name of the user
<display_name>	Display name to easily identify the user

Example

```
set system user smith display-name primeuser
```

Enabling or disabling CLI access

You can control a user’s access to the CLI. By default, users will have access to the CLI.

set system user <username> **no-cli-access** < **true** | **false** >

Command arguments

<username>	Name of the user
<true | false>	When the no-cli-access is set to true, the user’s access to the CLI is disabled. When the no-cli-access is set to false, the user will be able access the CLI.

Example

```
set system user smith no-cli-access false
```

Configuring sudo Without Password

You can configure local system users to log in via passwords or using SSH keys. From a security perspective, it is desirable to allow authentication with SSH keys only. RBFS provides a configuration knob to disable the requirement for a 'sudo' password so that local users can authenticate with SSH keys only. This knob is configurable only if the user or one of its roles is supervisor.

You can enter the following command to enable or disable the 'sudo' password. By default, this is set to false which ensures that the supervisor must provide a password when using sudo.

set system user <user> **no-sudo-password** < **true** | **false** >

Command arguments

<username>	Name of the user
<true | false>	When the no-sudo-password is set to true, it indicates that a 'sudo' password is not required. When it is set to false, it indicates that the supervisor must provide a password when using sudo.

Example Configuration:

```
{
    "rtbrick-config:user": [
      {
        "username": "smith",
        "role": "supervisor",
        "shell": "/bin/bash",
        "ssh-pub-key": "ssh-rsa AAAAB3Nza<...>",
        "no-sudo-password": "true"
      }
    ]
 }
```

If 'no-sudo-password' is set, you can log in with your SSH key.

Configuring Fail2Ban

The failed SSH login attempts from one user over an SSH jump host affect other users from the same jump host. You can configure Fail2Ban, which enables you to whitelist some IP addresses by creating separate jails for each user connecting through the jump host. In this way, failed login attempts by one user will not affect another user.

The following Fail2Ban command allows you to configure a list of IP addresses to include those IP addresses in the whitelist. You can specify multiple IP addresses that you want to exclude from the ban. Fail2Ban is applicable to both the ONL and the Linux container.

Syntax:

set system platform-management fail2ban ignore-ip <ignore-ip>

Command arguments

ignore-ip <ignore-ip>	Specify the IP addresses in CIDR notation which are to be whitelisted.

Example commands for configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config set
set system
set system platform-management fail2ban ignore-ip 10.1.1.1/32
set system platform-management fail2ban ignore-ip 10.1.1.2/32
set system platform-management fail2ban ignore-ip 10.2.2.0/24
```

Example Configuration:

```
supervisor@rtbrick>LEAF01: cfg> show config
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "platform-management": {
        "fail2ban": {
          "ignore-ip": [
            "10.1.1.1/32",
            "10.1.1.2/32",
            "10.2.2.0/24"
            ]
        }
      }
    }
  }
}
```

Enabling or Disabling Password-based Authentication

The "set system authentication ssh" CLI allows modifying the SSH configurations on the device. The password-authentication option disables the password-based authentication for user login. By default, password-based authentication is enabled. Additionally, two more SSH authentication parameters can be configured: client-alive-count-max and client-alive-interval.

After disabling password-based authentication, the device will not be accessible through password-based login; hence, you must configure a public key for the user before disabling this. If you have disabled password authentication by mistake and forgot to configure the public key, you can only recover the device using RESTCONF by configuring the password authentication to “true.”

Syntax:

set system authentication ssh <attribute> <value>

Attribute	Description
client-alive-count-max	Defines the maximum number of unanswered keepalive messages before terminating the connection.
client-alive-interval	Sets the interval (in seconds) for sending keepalive messages to the client to prevent idle disconnections.
password-authentication	Enables or disables password-based authentication, set to 'false' to enforce key-based authentication for enhanced security.

The example below shows disabling password-based authentication and logging in using SSH key.

```
supervisor@rtbrick: cfg> set system authentication ssh password-authentication false
supervisor@rtbrick: cfg> commit
supervisor@rtbrick: cfg> set system user supervisor
supervisor@rtbrick: cfg> set system user supervisor shell /bin/bash
supervisor@rtbrick: cfg> set system user supervisor ssh-pub-key "ssh-ed25519
ABCDEFC1lZDI1NTE5AAcRsYHrrTesQr04UuSrtefN/ndezZM7+8mBC"
supervisor@rtbrick: cfg> commit
supervisor@rtbrick: cfg>
```

Configuration Example:

```
{
    "ietf-restconf:data": {
        "rtbrick-config:system": {
            "authentication": {
                "ssh": {
                    "password-authentication": "false"
                }
            },
            "user": [
                {
                    "username": "supervisor",
                    "shell": "/bin/bash",
                    "ssh-pub-key": [
                        "ssh-ed25519 ABCDEFC1lZDI1NTE5AAcRsYHrrTesQr04UuSrtefN/ndezZM7+8mBC"
                    ]
                }
            ]
        }
    }
}
```

The example below shows how to log into the device after disabling password-based authentication.

```
supervisor@rtb:~$ ssh supervisor@bl1-pod1
Last login: Wed Feb 26 07:28:06 2025 from 192.168.1.135
supervisor@rtb:~$ ssh supervisor@bl1-pod1
Last login: Wed Feb 26 07:28:06 2025 from 192.168.1.135

+--------------------------------------------------------------------+
|                                                                    |
    RBFS container rtbrick running on bl1-pod1:
        Date:       Wed Feb 26 07:29:06 AM UTC 2025
        Uptime:     up 1 hour, 25 minutes

    Image metadata:
        UUID:       0d5b04c0-7b91-4a42-bc5b-80e879bddcf2
        Version:    25.1.1-pre-release.8
        Role:       l2bsa
        Platform:   qax
        Model:      s9500-22xst
        Format:     lxd
        Build date: 2025-02-25 09:25:40 UTC
        Based on:   Ubuntu 22.04.5 LTS

    RBFS License:
        No license found
|                                                                    |
+--------------------------------------------------------------------+

supervisor@rtbrick>bl1-pod1:~ $
```

The example below show how to re-enable the password-based authentication and log in to the device.

```
supervisor@rtbrick: cfg> set system authentication ssh password-authentication true
supervisor@rtbrick: cfg> commit
supervisor@rtbrick: cfg> delete system user supervisor
supervisor@rtbrick: cfg> commit
supervisor@rtbrick: cfg>
```

Configuration Example:

```
{
    "ietf-restconf:data": {
        "rtbrick-config:system": {
            "authentication": {
                "ssh": {
                    "password-authentication": "true"
                }
            }
        }
    }
}
```

The example below shows how to log into the device after enabling password-based authentication.

```
supervisor@rtb:~$ ssh supervisor@bl1-pod1
supervisor@bl1-pod1's password:
Last login: Wed Feb 26 07:29:06 2025 from 192.168.1.135

+---------------------------------------------------------------------+
|                                                                     |
    RBFS container rtbrick running on bl1-pod1:
        Date:       Wed Feb 26 07:30:41 AM UTC 2025
        Uptime:     up 1 hour, 27 minutes

    Image metadata:
        UUID:       0d5b04c0-7b91-4a42-bc5b-80e879bddcf2
        Version:    25.1.1-pre-release.8
        Role:       l2bsa
        Platform:   qax
        Model:      s9500-22xst
        Format:     lxd
        Build date: 2025-02-25 09:25:40 UTC
        Based on:   Ubuntu 22.04.5 LTS

    RBFS License:
        No license found
|                                                                     |
+---------------------------------------------------------------------+

supervisor@rtbrick>bl1-pod1:~ $
```

15.4. sFlow

15.4.1. RBFS sFlow Technology

sFlow Overview

sFlow is a packet-sampling protocol for monitoring network traffic. sFlow enables continuous, real-time monitoring of network traffic across all interfaces simultaneously. It works by collecting samples of data packets and transmitting these samples in UDP datagrams to a central station called collector.

The sampling process is directly handled by the networking processing ASIC, which ensures accuracy in packet sampling and also significantly reduces the utilization of control-plane resources such as CPU, memory.

sFlow provides insights into network traffic that enables the network operator to promptly identify performance issues, security threats, or unusual traffic patterns. It allows real-time network monitoring across thousands of router ports simultaneously in a large scale environment.

Packet-Based Sampling:

Packet-based sampling is a technique for capturing a subset of network traffic by selecting one packet from a predefined number of packets. It generates a representative sample of the overall traffic without capturing every packet, which would be resource-intensive.

RBFS sFlow mechanism captures only the packet headers passing through the incoming and outgoing physical interfaces.

Benefits of sFlow Technology

sFlow offers the following benefits:

Detect Network Issues and Manage Real-time Congestion: sFlow provides visibility into traffic patterns and enables the detection of anomalies such as misconfigurations, packet loss, and latency and it can track unauthorized activities. It can monitor traffic flow and bandwidth usage in real-time and provides data to manage network congestion.

For example, if an RBFS device suddenly experiences high latency or packet loss, sFlow provides information to pinpoint the source of the issue, such as a faulty interface or a congested port.

Types of Applications and Usage Patterns: By analyzing packet headers, sFlow can identify different types of network traffic such as web browsing, peer-to-peer (P2P), and DNS queries. This helps you to understand the types of applications and the amount of bandwidth that they consume. It allows to track changes in resource usage.

For example, if a network experiences an increase in P2P traffic, sFlow can provide the cause and the users or applications that are involved. With this data, network administrators can take appropriate actions.

Billing and Charge-Back: sFlow can provide resource usage data on per-application or per-user basis, which is useful for enterprise customers that require billing or internal charge-back. Enterprises can utilize this data to accurately bill customers or to allocate costs to internal departments based on the actual network consumption.

Route Profiling and Peering Optimization: sFlow captures detailed routing information and traffic distribution that helps to analyze traffic patterns across different routes and optimize peering agreements. It helps to refine routing policies and peering relationships.

For example, service providers can use sFlow data to determine which peering partners receive most of the traffic. This information allows you to modify their peering policies to improve performance.

Trends and Capacity Planning: sFlow can provide data on traffic trends and bandwidth usage that helps you to increase capacity.

For example, you can see a particular link is nearing its bandwidth capacity by analyzing sFlow data. With this data, you can plan an upgrade.

Understanding RBFS sFlow Implementation

sFlow works by capturing sample network packets and collecting data periodically to give a representative view of the overall traffic. It does not capture every packet, instead, it samples a subset of packets. For example, it captures one packet in every N packets (where N is configurable) on a physical interface.

sFlow Components

An RBFS sFlow system comprises an sFlow agent embedded in a device and a remote sFlow collector.

sFlow Agent

The sFlow agent gathers data by sampling packets from the ports which are configured for sFlow. It then encapsulates this data into UDP datagram and sends the samples to the collector. The sFlow collector analyzes these samples to give you data about traffic patterns, issues, and performance of the network.

sFlow Agent is a key component and performs the following tasks:

1. Sampling to collect packet samples based on the defined number of packets
2. Process the collected packets
3. Export the UDP datagram to the sFlow collector

For sending the sampled packets, you can configure a destination port. The default destination port is 6343.

sFlow Collector

The collector receives the sampled data from the sFlow agents. It aggregates, processes, and stores the data for analysis. The collector can analyze and interpret metrics such as packet types, bandwidth usage, IP addresses, protocol information, and so on.

Flow Sampling and Number of Samples

With the packet sampling, an sFlow agent samples packets on a designated interface based on a sampling rate that is set.

Each sFlow packet carries sampled data, and the number of samples determines the amount of data included in a single packet. In RBFS, the number of samples in every sFlow packet by default is 1. It means each sFlow packet contains one sampled record.

Guidelines for sFlow Settings

- To successfully transport sFlow data to the collector, make sure the routing table includes a route that enables communication with the collector’s configured IPv4 address. Specifically, there should be a route entry for the collector’s IPv4 address within the instance where the sFlow collector is configured. This ensures that the device knows the correct path to reach the collector’s IP address within the specified routing instance, allowing sFlow data transit to the collector.
- The agent ID must be identical across all collectors configurations within a device. This ensures the source of sampled data is recognized as the same across multiple collectors.
- It is required to enable sFlow on each interface individually by attaching a traffic-sampling profile. This profile defines the collector IP address, sampling direction and so on.
- You can configure a maximum number of four different collectors to receive sampled traffic.

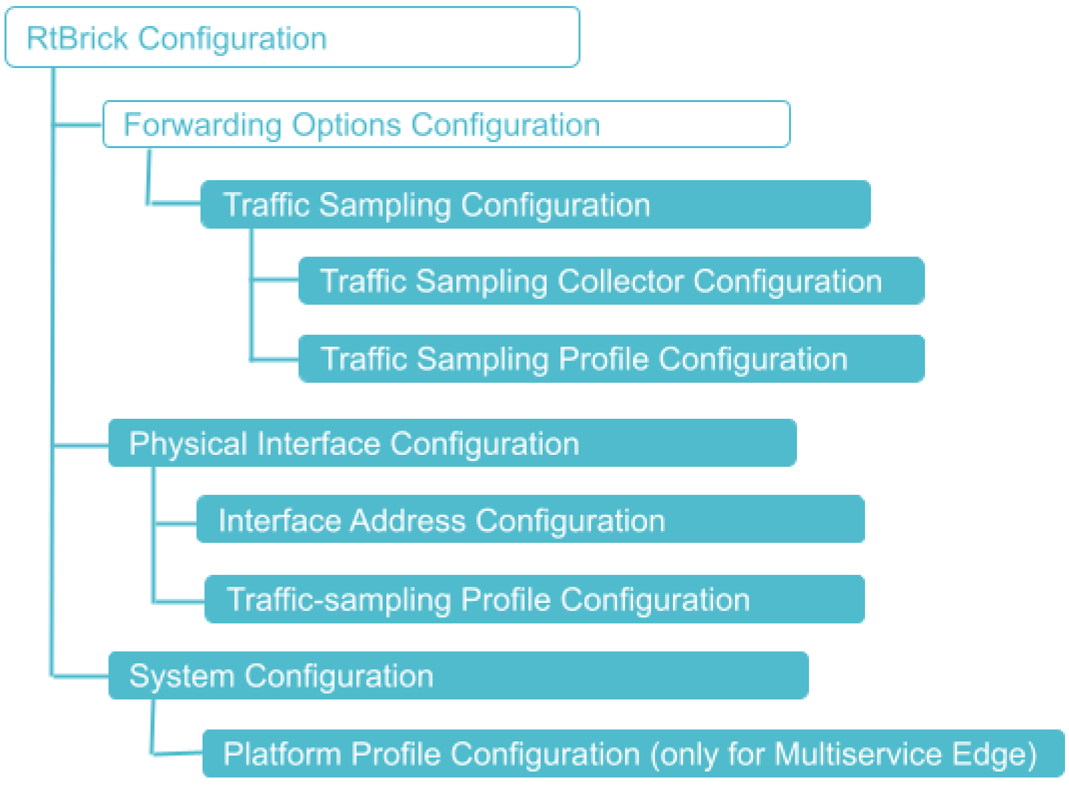
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

15.4.2. sFlow Configuration

Configuration Hierarchy

The following diagram illustrates the sFlow configuration hierarchy. In this configuration hierarchy, traffic sampling collectors and traffic sampling profiles are set up under forwarding-options to enable packet sampling using the sFlow protocol. After defining the collectors and sampling profiles, the sampling profile can be attached to the physical interfaces that allows the device to sample traffic from that interfaces.


RtBrick Configuration
Forwarding Options Configuration
Traffic Sampling Configuration
Traffic Sampling Collector Configuration
Traffic Sampling Profile Configuration
Physical Interface Configuration
Interface Address Configuration
Traffic-sampling Profile Configuration
System Configuration
Platform Profile Configuration (only for Multiservice Edge)

Configuration Syntax and Commands

The following sections describe the sFlow configuration syntax and commands.

Platform Profile Configuration

Platform Profile Configuration is applicable only to Multiservice Edge devices. On Spine devices, Platform Profile configuration is not required.

> The device must be rebooted after completing the Platform Profile configuration for it to take effect.

Syntax

set system platform profile <profile-name>

Attribute	Description
profile_name	Name of platform profile
feature	Feature name.

Example commands:

```
set system platform profile nat_4q
set system platform profile nat_4q feature sFlow
```

```
supervisor@rtbrick.net: op> show config system platform
{
  "rtbrick-config:platform": {
    "profile": [
      {
        "profile_name": "nat_4q",
        "feature": [
          "sFlow"
          ]
      }
    ]
  }
}
```

sFlow Collector Configuration

You can configure a maximum number of four collectors to receive sampled traffic.

> The agent ID must be identical across all collectors. This ensures the source of sampled data is recognized as the same across multiple collectors.

You can create, modify, or delete collectors dynamically, while the device continues to operate. This gives you the flexibility to modify the configurations of the sFlow collectors without causing any downtime.

Syntax

set forwarding-options traffic-sampling collector <collector-id> <attribute> <value>

Example commands:

```
set forwarding-options traffic-sampling collector 1
set forwarding-options traffic-sampling collector 1 instance default
set forwarding-options traffic-sampling collector 1 export-protocol sflow
set forwarding-options traffic-sampling collector 1 address ipv4 10.1.1.1
set forwarding-options traffic-sampling collector 1 sampling-rate 100
set forwarding-options traffic-sampling collector 1 destination-udp-port 6343
set forwarding-options traffic-sampling collector 1 agent-id 1.2.3.4
```

Attribute	Description
collector-id	Identifier for each collector.
agent-id	Unique identifier for the device sending sFlow data. This ID must be the same across all collectors.
collector-ip	The IP address of the collector to which the sFlow UDP data gram is sent.
instance	The Instance of collector IP reachability.
destination-udp-port	The UDP port used for sending sFlow data to the collector (Default: 6343).
sampling-rate	Specifies the rate at which the packets are sampled. Range: 100 - 50,000. Example: When configured as 100, 1 packet out of 100 packets gets sampled.
export-protocol	The protocol for sending the captured sample packets to the collector (Protocol used is sFlow).

Example: Collector configuration

The following configuration sets up sFlow collectors. Each collector is assigned with a unique ID and is configured with an IP address to receive the sampled traffic, along with parameters like sampling rate, number of samples, and the destination UDP port for sFlow data. The same agent ID (1.2.3.4) is used for all collectors. The sampled traffic is sent to the collector’s IP address, 10.1.1.1, over UDP port 6343. All collectors operate within the 'default' instance.

Collector 1 is configured with a sampling rate of '1' that indicates every packet traversing the configured interface is sampled without skipping any packet. The number of samples specified is 3, which indicates that the individual packets are aggregated into groups of three samples before being exported to the collector. The sFlow agent, which is the device or system sending the traffic samples, is defined by the agent identifier ‘1.2.3.4’.

In this configuration, collectors 2, 3, and 4 sample every 10th packet but differ in the number of samples and destination ports. Agent ID 1.2.3.4 is configured for all collectors 2, 3, and 4.

```
{
    "rtbrick-config:forwarding-options": {
        "traffic-sampling": {
            "collector": [
                {
                    "collector-id": 1,
                    "instance": "default",
                    "export-protocol": "sflow",
                    "address": {
                        "ipv4": "10.1.1.1"
                    },
                    "sampling-rate": 100,
                    "destination-udp-port": 6343,
                    "agent-id": "1.2.3.4"
                },
                {
                    "collector-id": 2,
                    "instance": "default",
                    "export-protocol": "sflow",
                    "address": {
                        "ipv4": "20.1.1.1"
                    },
                    "interval": 4,
                    "sampling-rate": 100,
                    "destination-udp-port": 6343,
                    "agent-id": "1.2.3.4"
                },
                {
                    "collector-id": 3,
                    "instance": "default",
                    "export-protocol": "sflow",
                    "address": {
                        "ipv4": "30.1.1.1"
                    },
                    "sampling-rate": 100,
                    "destination-udp-port": 6343,
                    "agent-id": "1.2.3.4"
                },
                {
                    "collector-id": 4,
                    "instance": "default",
                    "export-protocol": "sflow",
                    "address": {
                        "ipv4": "40.1.1.1"
                    },
                    "sampling-rate": 100,
                    "destination-udp-port": 6343,
                    "agent-id": "1.2.3.4"
                }
            ]
        }
    }
}
```

Traffic-Sampling Profile Configuration

You can create a traffic sampling profile that can be attached to the physical interface .

Syntax

set forwarding-options traffic-sampling profile <profile-name> **collector-id** <collector-id> **description** <description> **direction** [ingress\|egress\|both]

Example set of commands:

```
set forwarding-options traffic-sampling profile test
set forwarding-options traffic-sampling profile test collector 1
set forwarding-options traffic-sampling profile test direction ingress
set forwarding-options traffic-sampling profile test export-protocol sflow
```

Attribute	Description
profile-name	Name of the sampling profile.
collector-id	The identifier of the collector to which the packet samples are sent.
direction	Defines whether traffic sampling is performed on ingress, egress, or both of the directions.
description	Description for the profile.

Example: Traffic sampling profile configuration

This configuration defines two traffic sampling profiles that control how network traffic is sampled and exported to specific collectors using the sFlow protocol. Each profile focuses on capturing only incoming traffic on the monitored interfaces and exports this data to a designated collector for analysis.

The first profile, named "test", is configured to send the collected traffic data to Collector 1. This profile is designed to focus on incoming traffic.

The second profile, named "test1", has a similar configuration. It also monitors only incoming traffic, using the sFlow protocol to export the data. The traffic collected under this profile is directed to Collector 2.

```
{
    "rtbrick-config:forwarding-options": {
        "traffic-sampling": {
            "profile": [
                {
                    "profile-name": "test",
                    "collector": 1,
                    "direction": "ingress",
                },
                {
                    "profile-name": "test1",
                    "collector": 2,
                    "direction": "ingress",
                }
            ]
        }
    }
}
```

Attach Traffic-Sampling Profile with the Physical Interface

You can attach a traffic sampling profile to a physical interface using the following command.

Syntax

set interface <interface-name> **traffic-sampling-profile** <profile-name>

Example Command:

```
set interface ifp-0/1/10 traffic-sampling-profile test1
```

Attribute	Description
interface-name	Name of the physical interface.
<profile-name>	Name of the profile.

Example: Traffic sampling profile is attached on multiple physical interfaces.

This configuration sets up a few physical interfaces with different speeds. Some of these interfaces have traffic sampling profiles configured. Two interfaces (ifp-0/1/10 and ifp-0/1/8) are configured with traffic sampling profiles ('test1' and 'test', respectively).

```
[
    {
        "name": "ifp-0/1/10",
        "speed": "10G",
        "traffic-sampling-profile": "test1",
        "unit": [
            {
                "unit-id": 1,
                "address": {
                    "ipv4": [
                        {
                            "prefix4": "121.0.0.1/24"
                        }
                    ],
                    "ipv6": [
                        {
                            "prefix6": "fc66:1337:7331::1/64"
                        }
                    ]
                }
            },
            {
                "unit-id": 25,
                "vlan": 25,
                "address": {
                    "ipv4": [
                        {
                            "prefix4": "10.1.1.2/24"
                        }
                    ]
                }
            },
            {
                "unit-id": 26,
                "vlan": 26,
                "address": {
                    "ipv4": [
                        {
                            "prefix4": "20.1.1.2/24"
                        }
                    ]
                }
            }
        ]
    },
    {
        "name": "ifp-0/1/20",
        "speed": "10G"
    },
    {
        "name": "ifp-0/1/26",
        "speed": "1G"
    },
    {
        "name": "ifp-0/1/5",
        "speed": "1G"
    },
    {
        "name": "ifp-0/1/6",
        "speed": "1G"
    },
    {
        "name": "ifp-0/1/61",
        "speed": "10G"
    },
    {
        "name": "ifp-0/1/63",
        "speed": "10G"
    },
    {
        "name": "ifp-0/1/8",
        "speed": "10G",
        "traffic-sampling-profile": "test"
    },
    {
        "name": "ifp-0/1/9",
        "speed": "10G"
    },
    {
        "name": "lo-0/0/4",
        "unit": [
            {
                "unit-id": 1,
                "address": {
                    "ipv4": [
                        {
                            "prefix4": "1.2.3.4/24"
                        }
                    ]
                }
            }
        ]
    }
]
```

15.4.3. sFlow Operational Commands

sFlow Collector Information

The show traffic-sampling sflow collector command output provide details about the sFlow collectors configured on the device. It shows the information related to each collector, including its sampling rates, associated agent, destination IP address, and other configuration specifics.

Syntax:

show traffic-sampling sflow profile <profile-name> **collector** <collector-id>

Option	Description
<profile-name>	Name of the profile.
collector-id	Identifier for the collector.

Example 1: Summary Output for sFlow collectors

```
supervisor@rtbrick: op> show traffic-sampling sflow collector
Collector ID: 1
    Agent-ID: 1.2.3.4
    Address: 10.1.1.1
    Instance: default
    Number of samples: 5
    Sampling-rate: 10
    UDP-Destination-Port: 6343
Collector ID: 2
    Agent-ID: 1.2.3.4
    Address: 20.1.1.1
    Instance: default
    Sampling-rate: 10
    UDP-Destination-Port: 6343
    Interval: 4 secs
Collector ID: 3
    Agent-ID: 1.2.3.4
    Address: 30.1.1.1
    Instance: default
    Number of samples: 3
    Sampling-rate: 10
    UDP-Destination-Port: 6344
Collector ID: 4
    Agent-ID: 1.2.3.4
    Address: 40.1.1.1
    Instance: default
    Number of samples: 2
    Sampling-rate: 10
    UDP-Destination-Port: 6344
```

Example 2: Summary Output for a specified sFlow collector

```
supervisor@rtbrick: op> show traffic-sampling sflow collector 1
Collector ID: 1
    Agent-ID: 1.2.3.4
    Address: 10.1.1.1
    Instance: default
    Number of samples: 5
    Sampling-rate: 10
    UDP-Destination-Port: 6343
```

Sampling Profiles

Displays the packet sampling profile related information.

Syntax:

show traffic-sampling profile <options>

Option	Description
<profile-name>	Name of the profile.

Example 1: Summary Output for traffic sampling profiles

```
supervisor@rtbrick: op> show traffic-sampling sflow profile
Profile name: test
    Collector Id: 1
    Direction: ingress
Profile name: test1
    Collector Id: 2
    Direction: ingress
```

Example 2: Summary Output for traffic sampling for a specified profile

```
supervisor@rtbrick: op> show traffic-sampling sflow profile test
Profile name: test
    Collector Id: 1
    Direction: ingress
```

16. Telemetry

16.1. TSDB

16.1.1. Prometheus Time Series Database (TSDB) Integration Overview

Operational-state visibility is key for troubleshooting, testing, monitoring and capacity management. This requires to sample router metrics periodically. Ingestion of time-series data allows to ask of interesting operational queries.

Examples:

- A slightly increasing memory consumption over time while the overall PPPoE session count has not changed, for example, is an indication of a memory leak.
- If the 5-minute chassis temperature is too high, this might be an indication for an imminent hardware breakdown and the switch hardware must be replaced.
- If utilization of all fabric interfaces is constantly touching the 80% saturation levels then new fabric links must be commissioned.
- High input traffic with degradation of optical receive levels might be an indication of running very close to the optical budget.

The challenge is to sample all this information efficiently in terms of disk, memory and CPU utilization while providing comprehensive query and reporting functionality.

IP Pool Monitoring and Export

The IP Pool Monitoring and Export functionality enables real-time insights into IP allocation and enables more efficient resource management. It provides 'usage' and 'size' attributes that allows for accurate monitoring IP pool utilization through TSDB and supports continuous export via Prometheus.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

Architectural Overview

The RBFS telemetry architecture is based on Prometheus as an open-source systems monitoring and alerting toolkit. Prometheus is designed to pull metrics periodically and save them efficiently. It allows us to analyze the metrics with a powerful query language called PromQL. Also, an optional alert management is available. There is an opportunity to tie it together with its own services to integrate it into the system landscape. Data should have short retention times (default 15d).

This fits perfectly with the needs of BDS. The figure below shows how it fits in an overall architecture.

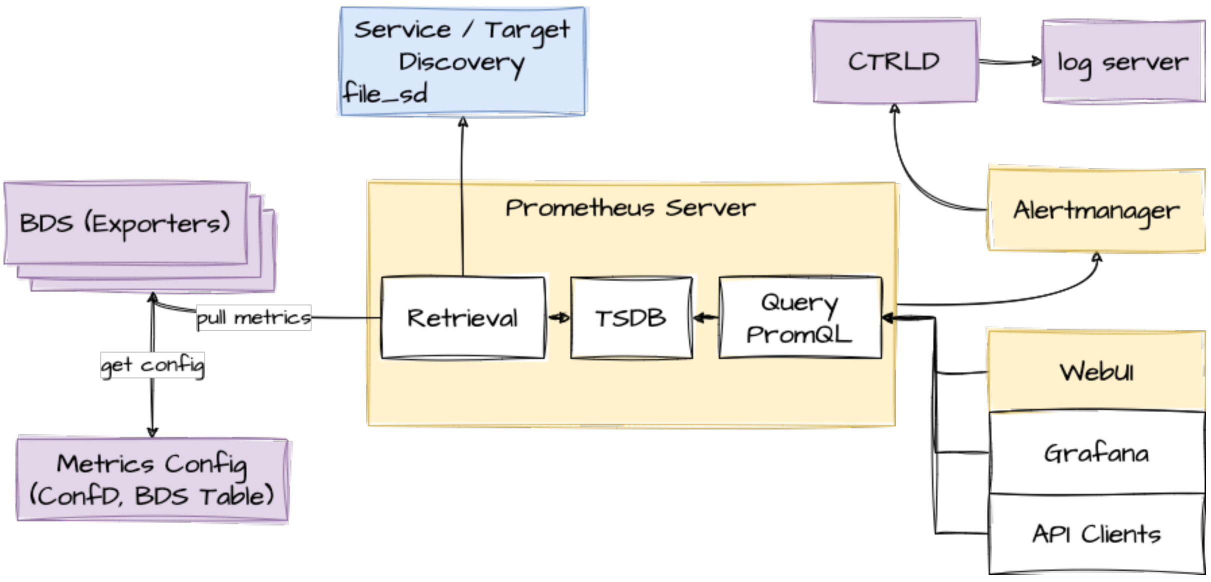


Figure 27. Prometheus in RBFS

To mitigate the short retention times, which fits to BDS but not in an overall telemetry process, the data can be stored in a centralized storage database (for example, Influx) this can be done by federation or via remote storage adapters. To distribute the alert messages from Prometheus, CTRLD functions as an "alertmanager webhook receiver", which takes the alert and distributes it to a log management tool.

Router deployment model

Prometheus DB is run on the router as a dedicated process. It ships with a package-time configuration to poll each BDS-capable speaker at periodic intervals. Initially the periodic interval is 1 second. The Prometheus Exposition format is a very simple HTTP-based GET query that asks a given BD speaker "Give me all your metrics". Each BD subscribes to the *global.time-series.metric.config* table, which contains an operator-configurable list of BDS targets. Only the BDS which is the master of a table responds. Next Prometheus polls the BD using the */metrics* URL.

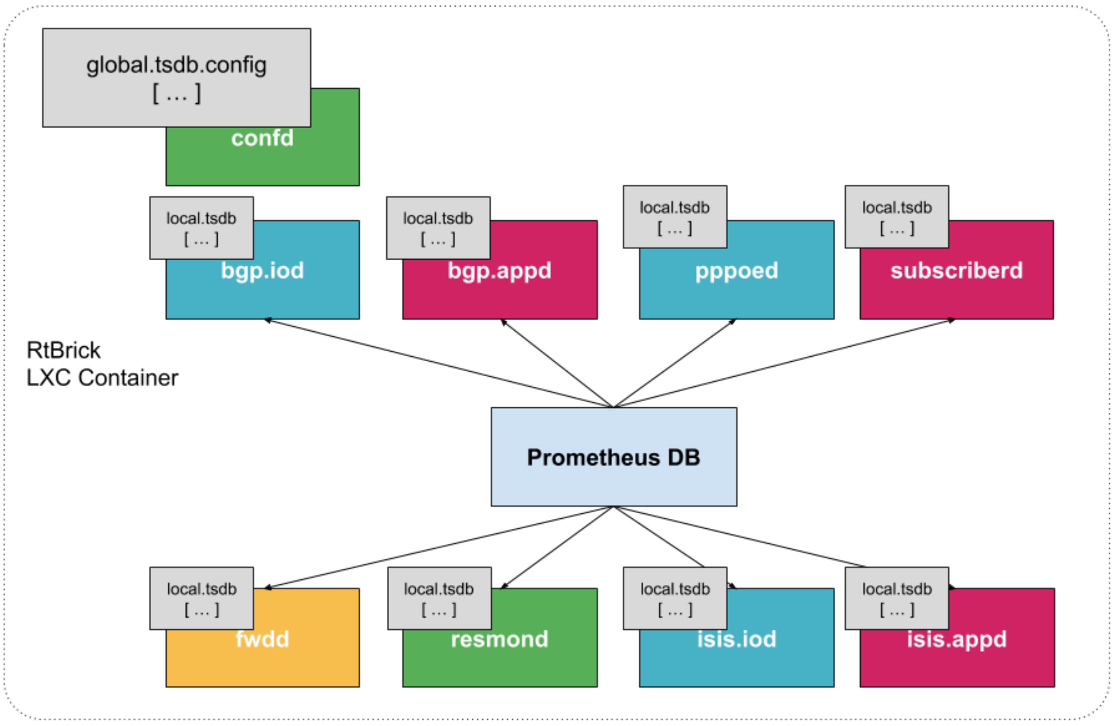


Figure 28. Prometheus in RBFS with the different scrape target

Storage efficiency

On average Prometheus uses only around 1-2 bytes per sample. Thus, to plan the capacity of a Prometheus server, you can use the rough formula:

```
needed_disk_space = retention_time_seconds * ingested_samples_per_second *
bytes_per_sample
```

The single binaries disk space:

```
-rwxr-xr-x 1 root root 27M Sep 2 22:51 alertmanager +
-rwxr-xr-x 1 root root 81M Sep 2 22:51 prometheus +
-rwxr-xr-x 1 root root 49M Sep 3 19:55 promtool
```

Promtool is needed to test the configurations before set them to prometheus.

Alerting

The alerting is configured through Prometheus. For more information, see alertmanager.

Role of CTRLD

Prometheus and Alertmanager register themselves in CTRLD, so that CTRLD is aware of these two services.

Refer to Figure-1, which provides an overview of the role of CTRLD.

Service state and Proxy

The registration of the services gives 2 advantages:

1. The operational state is an indicator that the service is up and running.
2. The proxy functionality of CTRLD can be used for Prometheus and alertmanager.

The proxy functionality is used for querying Prometheus directly:

```
curl
'http://198.51.100.125:19091/api/v1/rbfs/elements/rtbrick/services/prometheus/prox
y/api/v1/query?query=up' | jq .
```

But it is also used for federation and therefore the following URL is used:

```
http://198.51.100.125:19091/api/v1/rbfs/elements/rtbrick/services/prometheus/proxy
/federate
```

Alert distribution

CTRLD can forward the alerts from the alertmanager to Graylog or any other REST endpoint.

API for Configuration

CTRLD provides a REST API Endpoint for the configuration of alerts and metrics.

Federation deployment model

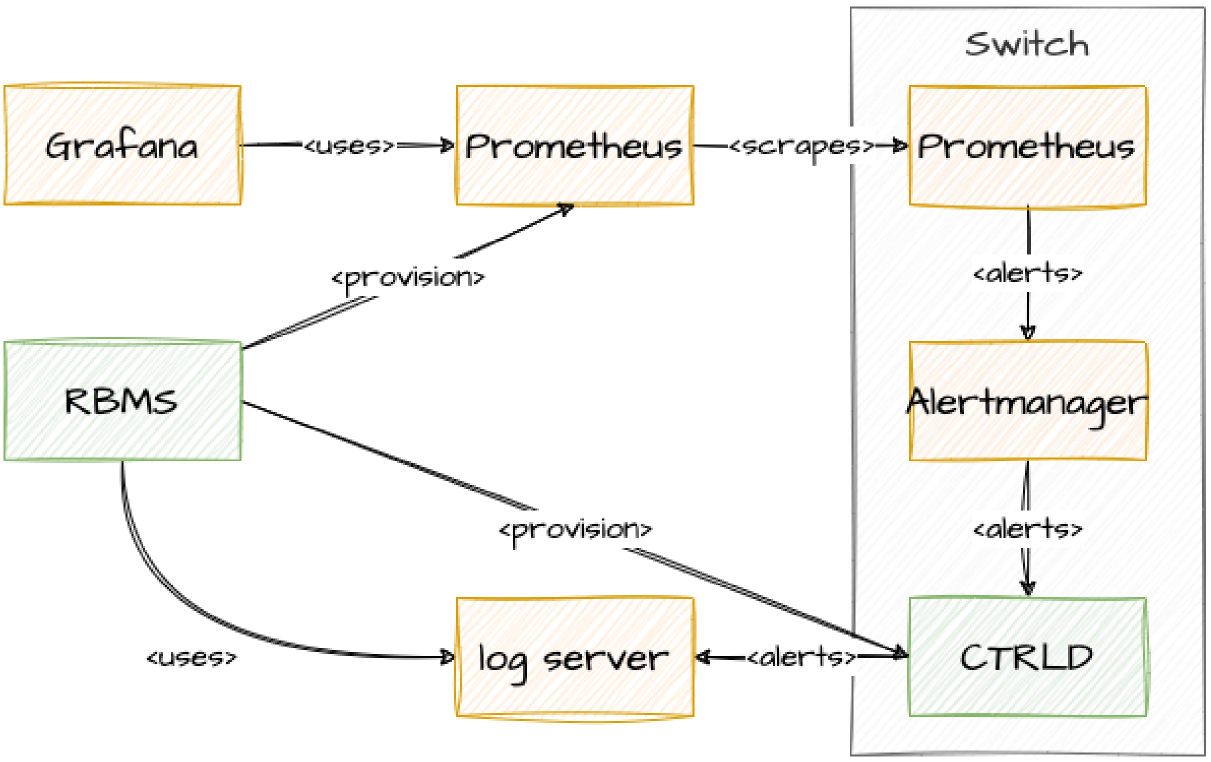


Figure 29. Federation of Prometheus, Alertmanager and graylog target

Prometheus is intended to have at least one instance per datacenter usually; also with a global Prometheus for global graphing or alerting. Federation allows for pulling metrics and aggregations up the hierarchy.

In the global Prometheus config, this time-series is pulled:

prometheus.yml:

```
global:
  scrape_interval: 60s # By default, scrape targets every 15 seconds.
  # A scrape configuration containing exactly one endpoint to scrape:
scrape_configs:
  - job_name: "federate"

    honor_labels: true
    metrics_path: '/federate'
    params:
      'match[]':
        - '{job="bds"}'
    scrape_interval: 15s
    # Patterns for files from which target groups are extracted.
    file_sd_configs:
      - files:
          - ./bds.target.yml
        refresh_interval: 5m
```

The match[] here requests all BDS job time series. By following this job naming convention, you do not have to adjust the config whenever there is a new aggregating rule.

The targets itself can be configured in a separate file.

bds.target.yml:

```
- targets: ['198.51.100.125:19091']
  labels:
    __metrics_path__:
"/api/v1/rbfs/elements/rtbrick/services/prometheus/proxy/federate"
    box: 125_rtbrick
```

16.1.2. TSDB Configuration

The following section describes how to configure the system to gather metrics and alerts out of the system.

Metric

To better understand the Data Model have a look at the Prometheus Data Model.

Metric Data Model

In RBFS it is possible to turn each table attribute into a metric.

When you export the time-series metric data for an attribute which has more than 50 label values (user-defined, default labels), you may see truncated data in the exported metric.

The following table describes the configuration model:

Metric	
metric_name	Name of the metric (metric name conventions). That is the unique identifier for the metric.
table_name	Table Name for which the metric is designed, could also be a regular expression.
append_timestamp	Timestamp is epoch rendered in milliseconds and its value is equal to current metrics value’s creation time in RBFS.

bds_metric_type	• object-metric: if the metric should be gathered from regular table attributes • index-metric: if the metric should be gathered out of an attribute of an index table
index_name	Name of the index, if the bds_metric_type is index-metric.
metric_type	• gauge: is a metric that represents a single numerical value that can arbitrarily go up and down. Gauges are typically used for measured values like temperatures or current memory usage, but also "counts" that can go up and down, like the number of concurrent requests. • counter: is a cumulative metric that represents a single monotonically increasing counter whose value can only increase or be reset to zero on restart. For example, you can use a counter to represent the number of requests served, tasks completed, or errors. Do not use a counter to expose a value that can decrease. For example, do not use a counter for the number of currently running processes; instead use a gauge.
metric_description	Description of the metric.
attributes	List of Attributes (see Attribute Table) that will be streamed as metric.
filters	List of AttributeFilters (see AttributeFilter Table) that filters the table rows which should be considered for metric generation. Each filter in this list has to match in order to generate the metric, so the list implies an implicit AND.

Attribute	
attribute_name	Name of the attribute that should be streamed as metric. This Attribute has to be a numeric type, or a type that has a numeric converter.

filters	List of AttributeFilters (see the [tsdb:tsdb_config:::AttributeFilter] table) that filters the table rows which should be considered for metric generation. Each filter in this list has to match in order to generate the metric, so the list implies an implicit AND.
labels	List of AttributeLabels (see the [tsdb:tsdb_config:::AttributeLabel] table) that are attached to that metric.

AttributeFilter	
match_attribute_name	Attribute of the Table which is used to match against.
match_type	• exact: so the attribute has to match exactly the match value • regular-expression: the match value is a regular expression the attribute must match
match_value	The value that attribute has to match against.

AttributeLabel **CAUTION**: Remember that every unique combination of key-value label pairs represents a new time series, which can dramatically increase the amount of data stored. Do not use labels to store dimensions with high cardinality (many different label values), such as user IDs, email addresses, or other unbounded sets of values.	
label_name	Name of the Label (label name conventions).
dynamic	bool: If the label is dynamic, the label_value is treated as attribute_name, so the value of the attribute is used as the label value, otherwise the label value is used directly.
label_value	The value of the label or the attribute which should be used as label value.

filters	List of AttributeFilters (see [tsdb:tsdb_config:::AttributeFilter] Table) that filters the table rows which should be considered for label generation. Each filter in this list has to match in order to generate the label, so the list implies an implicit AND.

Configuring Metrics

The configuration of the Metrics can be done in various ways.

Configuring Metrics using Command Line Interface

To configure the Time Series Database, perform the following steps:

1. Define Metric configuration
2. Define Attribute configuration
3. Optional Filters at Metric Level and Attribute level
4. Defining labels to be attached to exported metric

Metric Configuration

Metric configuration is used to configure the parameters of the metric data being exported.

Depending on the platform the exact resource name to be monitored can be found in global.chassis_0.resource.sensor, and adjust the Prometheus/Grafana configuration accordingly.

Syntax

set time-series metric <name>

set time-series metric <name> **description** <128 character description about the metric-name>

set time-series metric <name> **prometheus-type** <counter / gauge>

set time-series metric <name> **bds-type** <object-metric / index-metric>

set time-series metric <name> **table-name** <table-name>

set time-series metric <name> **attribute** <attribute-name>

set time-series metric <name> **index-name** <index-name>

set time-series metric <name> **append-timestamp** <true>

set time-series metric <name> **filter** <match-attribute-name>

set time-series metric <name> **include-subscribed-tables** [**true** / **false**]>

Command arguments

<metric-name>	Specifies the name of the metric exported, as would be reflected in Prometheus. Use the naming conventions as recommended by Prometheus
<128 character description about the metric-name >	Description of the metric
<counter / gauge>	Configures the metric data type. Currently the supported Prometheus metric data are: counter and gauge
<object-metric / index-metric >	Specifies the type of attribute, that is scraped and exported. There are two types, object-metric and index-metric
<table-name>	Specifies the target table, from which the data is scraped and exported.
<attribute-name>	Specifies the name of the attribute, in the target table to be scraped and exported
<append-timestamp>	Set the append-timestamp to true for exporting the metric values with timestamp. By default, the value is 'false'.
<index-name>	Specifies the index-name of the index-metric attribute. This configuration is applicable for index-metric alone.

<match-attribute-name>	Specifies the matching attribute name for the filter
include-subscribed-tables [true / false]	Specifies whether the configuration needs to be applied on a subscribed tables as well. Default: false.

Example

```
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm table-name
global.chassis_0.resource.sensor
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm bds-type object-
metric
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm prometheus-type
gauge
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm description
"Chassis fan speed in rpm"
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm include-
subscribed-table false
```

Allowed Attribute Types (Type Converters)

Normally only attributes are allowed, which are of type numeric, but for some types, there are built-in type converters, which allow also to use attributes of their types.

For the following BDS types, built-in type converters are provided by BDS. As per Prometheus data model, type converter will convert the BDS type into a 64bit float number.

BDS data type	Outcome number represents
unix-wallclock-timestamp	Seconds
unix-usec-wallclock-timestamp	Seconds
unix-usec-monotonic-timestamp	Seconds
unix-usec-coarse-wallclock-timestamp	Seconds
bandwidth	bps(bit per second)
temperature	Degree Celsius

Metric Filter Configuration

Metric filter configuration is used to configure the parameters of the filter. It is used to filter the exported metric. This is an optional configuration.

Syntax

set time-series metric <name> **filter** <match-attribute-name>

set time-series metric <name> **filter** <match-attribute-name> **match-type** <exact / regular-expression>

set time-series metric <name> **filter** <match-attribute-name> **match-attribute-value** <match-attribute-value>

Command arguments

<match-attribute-name>	Specifies the filter that filters the exported metric, based on specified criteria. This is optional configuration.
< exact / regular-expression >	Specifies the match type to be used, There are two options, exact and regular-expression.
<match-attribute-value>	Specifies the attribute value used for match. Fixed value for exact. Regex pattern for regular-expression

Example

Exact Value

```
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm filter
resource_type match-attribute-value fan
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm filter
resource_type match-type exact
```

Regular Expression

```
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm filter
resource_name match-attribute-value Chassis.*
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm filter
resource_name match-type regular-expression
```

Metric Attribute Label Configuration

Metric attribute config is used to configure the labels to be attached to the exported metric.

Syntax

set time-series metric <name> **attribute** <attribute-name> **label** <label-name>

set time-series metric <name> **attribute** <attribute-name> **label** <label-name> **label-type** <dynamic / static>

set time-series metric <name> **attribute** <attribute-name> **label** <label-name> **label-value** <label-value>

Command arguments

<label-name>	Specifies the name of label. User definable, Please use naming conventions as recommended by Prometheus
<dynamic / static>	Specifies the type of labels, a static value or dynamic value to be added.
<label-value>	Specifies the label-value to be used.

Example

Dynamic Label

```
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
label fan
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
label fan label-value resource_name
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
```

```
label fan label-type dynamic
```

Static Label

```
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
label vender
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
label fan label-value rtbrick
admin@rtbrick: cfg> set time-series metric chassis_fan_speed_rpm attribute rpm
label fan label-type static
```

Metric Attribute Filter Configuration

Attribute filter config is used to configure the parameters of Attribute filter. It is used to filter the exported metric based on certain fields of the attribute. This is an optional configuration.

Syntax

set time-series metric <name> **attribute** <attribute-name> **filter** <match-attribute-name>

set time-series metric <name> **attribute** <attribute-name> **filter** <match-attribute-name> **match-type** <exact/regular-expression>

set time-series metric <name> **attribute** <attribute-name> **filter** <match-attribute-name> **match-value** <match-attribute-value>

Command arguments

<attribute name>	Specifies the filter that filters the exported metric , based on criteria of the attribute. This is optional config.
<exact / regular-expression>	Specifies the match type to be used, There are two options, exact and regular-expression.
<match-attribute-value>	Specifies the attribute value used for match. Fixed value for exact. Regex pattern for regular-expression

Example

The below example shows, the metric attribute will be exported only if the port_stat_if_in_discards is exactly 0.

```
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts filter port_stat_if_in_discards
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts filter port_stat_if_in_discards match-type exact
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts filter port_stat_if_in_discards match-attribute-value 0
```

Metric Label Filter Configuration

Label filter configuration is used to set filter parameters that can be used to attach label based on certain criteria. This is an optional configuration.

Syntax

set time-series metric <name> **attribute** <attribute-name> **label** <label-key> **filter** <match-attribute-name>

set time-series metric <name> **attribute** <attribute-name> **label** <label-key> **filter** <match-attribute-name> **match-type** <regular-expression/exact>

set time-series metric <name> **attribute** <attribute-name> **label** <label-key> **filter** <match-attribute-name> **match-attribute-value** <match-attribute-value>

Command arguments

<match-attribute-name>	Specifies the filter that filters the exported metric, based on some attribute value.This is optional config.
< exact / regular-expression >	Specifies the match type to be used, There are two options, exact and regular-expression.
<match-attribute-value>	Specifies the attribute value used for match. Fixed value for exact. Regex pattern for regular-expression

Example

The below example sets label, interface_orientation to the exported data, only if the interface_name matches ifp-0/0/50.

```
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts label interface_orientation
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts label interface_orientation filter interface_name
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts label interface_orientation filter interface_name
match-type exact
admin@rtbrick: cfg> set time-series metric interface_statistics_data attribute
port_stat_if_in_ucast_pkts label interface_orientation filter interface_name
match-attribute-value  ifp-0/0/50
```

Alert

RBFS uses the prometheus alerting feature to generate alerts. These alerts are forwarded to an alertmanager instance inside the rbfs container. The alertmanager instance sends the alert to CTRLD which distributes the alert to an HTTP Endpoint.

Alerts are also configured in a BDS table, and they are exported to Prometheus by the system.

Alert Data Model

Alert	
name	The name of the alert rule. That is the unique identifier for the rule.
group	Name of the alert group the alert belongs to. The alert group helps to structure the alerts.

interval	How often the rule should be evaluated. Pattern:"[0-9]+(ms |[smhdwy]" Example:"5s" In Prometheus the the interval can specified per alert group. So the alert alert group for Prometheus is calculated via {alert_group}_{interval}.
expr	Alert evaluation expression in promql
labels	Key, Value pairs of labels that should be applied. The labels clause allows specifying a set of additional labels to be attached to the alert. Any existing conflicting labels will be overwritten. The label values can be templated (see templating).
annotations	Key, Value pairs of annotations that should be applied. The annotations clause specifies a set of informational labels that can be used to store longer additional information such as alert descriptions or runbook links. The annotation values can be templated (see templating)
for	Alerts are considered firing once they have been returned for this long. Alerts which have not yet fired for long enough are considered pending. Pattern:"[0-9]+(ms |[smhdwy]" Example:"30s"
level	This is an explicit annotation label with the label name level. This is used to specify the severity: 1.Alert The annotation value can be templated (see templating)
summary	This is an explicit annotation label with the label name summary. The annotation values can be templated (see templating).

description	This is an explicit annotation label with the label name description. The annotation values can be templated (see templating).

Configuration

The configuration of the Metrics can be done in various ways.

Configuring Alert Using CLI

Syntax

```
set time-series alert <name>
set time-series alert <name> group <group>
set time-series alert <name> for <for>
set time-series alert <name> interval <interval>
set time-series alert <name> expr <expr>
set time-series alert <name> level <level>
set time-series alert <name> summary <summary>
set time-series alert <name> description <description>
set time-series alert <name> labels <label>
set time-series alert <name> annotations <annotations>
```

Command arguments

<name>	The name of the alert rule. That is the unique identifier for the rule.
<group>	Name of the alert group the alert belongs to. The alert group helps to structure the alerts.

<name>	**The name of the alert rule. That is the unique identifier for the rule.**
<interval>	How often the rule should be evaluated. Pattern:"[0-9]+(ms |[smhdwy]" Example:"5s" In Prometheus the the interval can specified per alert group. So the alert alert group for Prometheus is calculated via {alert_group}_{interval}.
<expr>	Alert evaluation expression in promql
<label>	Key, Value pairs of labels that should be applied. The labels clause allows specifying a set of additional labels to be attached to the alert. Any existing conflicting labels will be overwritten. The label values can be templated (see templating).
<annotations>	Key, Value pairs of annotations that should be applied. The annotations clause specifies a set of informational labels that can be used to store longer additional information such as alert descriptions or runbook links. The annotation values can be templated (see templating)
<for>	Alerts are considered firing once they have been returned for this long. Alerts which have not yet fired for long enough are considered pending. Pattern:"[0-9]+(ms |[smhdwy]" Example:"30s"
<level>	This is an explicit annotation label with the label name level. This is used to specify the severity: 1.Alert The annotation value can be templated (see templating)

<name>	**The name of the alert rule. That is the unique identifier for the rule.**
<summary>	This is an explicit annotation label with the label name summary. The annotation values can be templated (see templating).
<description>	This is an explicit annotation label with the label name description. The annotation values can be templated (see templating).

Example

```
admin@rtbrick: cfg> set time-series alert sample_alert
admin@rtbrick: cfg> set time-series alert sample_alert group hardware_metrics
admin@rtbrick: cfg> set time-series alert sample_alert for 30s
admin@rtbrick: cfg> set time-series alert sample_alert interval 5s
admin@rtbrick: cfg> set time-series alert sample_alert expr
avg_over_time(cpu_temperature_celcius[1m])>100
admin@rtbrick: cfg> set time-series alert sample_alert level 2
admin@rtbrick: cfg> set time-series alert sample_alert summary "Element {{
$labels.element_name }} CPU {{$labels.cpu}} HIGH temperature"
admin@rtbrick: cfg> set time-series alert sample_alert description "Cpu {{
$labels.cpu }} of element {{ $labels.element_name }} has a temperature o
ver 100 for more than 30 seconds"
admin@rtbrick: cfg> set time-series alert sample_alert labels device:leaf1
admin@rtbrick: cfg> set time-series alert sample_alert annotations "sample-
annotation-key:sample-value"
```

Enabling/Disabling Time Series Database History

In every Brick Daemon, the history of time series databases can be enabled. By default, time series database history is disabled.

Syntax

set time-series history-status <option>

Attribute	**Description**
[disable|enable]	Enable or disable time series database history. Time series database history is disabled, by default.

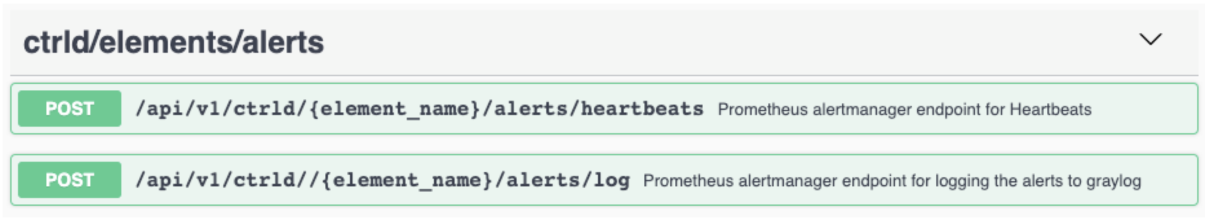
Example:

```
supervisor@S2-STD-7-7006>bm06-tst.fsn.rtbrick.net: cfg> show datastore confd table
global.time-series.config
```

```
Object: 0, Sequence 3, Last update: Mon May 23 09:02:28 GMT +0000 2022
  Attribute                                 Type                          Length
Value
  configuration_name (1)                    string (9)                         8
rtbrick
  time-series-history-enable (2)            boolean (6)                        1
False
```

Graylog Alert Distribution

The alertmanager on RBFS is configured to send alerts to CTRLD.



```
/api/v1/ctrld/{element_name}/alerts/heartbeats
```
Prometheus alertmanager endpoint for Heartbeats

```
/api/v1/ctrld//{element_name}/alerts/log
```
Prometheus alertmanager endpoint for logging the alerts to graylog

CTRLD therefore has an endpoint where the alerts are sent to. CTRLD translates the notification and forwards the message to the configured log management system. The instance used for forwarding is "prometheus".

Pool Metric Attributes Configuration

The following is an example for the new pool metric attributes:

IPv4 pool metric configuration commands:

```
set time-series metric pool_usage
set time-series metric pool_usage table-name local.access.1.ipv4.pool
set time-series metric pool_usage bds-type object-metric
set time-series metric pool_usage prometheus-type gauge
set time-series metric pool_usage description "IPv4 pool usage"
set time-series metric pool_usage attribute used
set time-series metric pool_usage attribute used label pool_name
set time-series metric pool_usage attribute used label pool_name label-value
pool_name
set time-series metric pool_usage attribute used label pool_name label-type
dynamic
set time-series metric pool_usage attribute used label pool_attribute
set time-series metric pool_usage attribute used label pool_attribute label-value
used
set time-series metric pool_usage attribute used label pool_attribute label-type
static
set time-series metric pool_usage attribute size
set time-series metric pool_usage attribute size label pool_name
set time-series metric pool_usage attribute size label pool_name label-value
pool_name
set time-series metric pool_usage attribute size label pool_name label-type
dynamic
set time-series metric pool_usage attribute size label pool_attribute
set time-series metric pool_usage attribute size label pool_attribute label-value
size
set time-series metric pool_usage attribute size label pool_attribute label-type
static
```

IPv6 pool metric Configuration commands:

```
set time-series metric ipv6_pool_usage
set time-series metric ipv6_pool_usage table-name local.access.1.ipv6.pool
set time-series metric ipv6_pool_usage bds-type object-metric
set time-series metric ipv6_pool_usage prometheus-type gauge
set time-series metric ipv6_pool_usage description "IPv6 pool usage"
set time-series metric ipv6_pool_usage attribute used
set time-series metric ipv6_pool_usage attribute used label pool_name
set time-series metric ipv6_pool_usage attribute used label pool_name label-value
pool_name
set time-series metric ipv6_pool_usage attribute used label pool_name label-type
dynamic
set time-series metric ipv6_pool_usage attribute used label pool_attribute
set time-series metric ipv6_pool_usage attribute used label pool_attribute label-
value used
set time-series metric ipv6_pool_usage attribute used label pool_attribute label-
type static
set time-series metric ipv6_pool_usage attribute size
set time-series metric ipv6_pool_usage attribute size label pool_name
set time-series metric ipv6_pool_usage attribute size label pool_name label-value
pool_name
set time-series metric ipv6_pool_usage attribute size label pool_name label-type
dynamic
set time-series metric ipv6_pool_usage attribute size label pool_attribute
set time-series metric ipv6_pool_usage attribute size label pool_attribute label-
value size
set time-series metric ipv6_pool_usage attribute size label pool_attribute label-
type static
```

16.1.3. Installation

The RtBrick fullstack comes with a ready to use tsdb instance. So no more installation on RBFS has to be done.

For federation of metrics, a global prometheus instance is needed. To visualize the metrics a Grafana instance has to be installed, and to get the alert messages, a graylog instance has to be set up. This document does not contain an installation guide for that systems.

The information about configuring a federation Prometheus to scrape metrics from a RBFS installation is described in the Federation deployment model section.

16.2. Resmon

16.2.1. Resource Monitoring (resmon) Overview

Monitoring the system resources is very crucial to analyze the health of devices. RBFS has a dedicated daemon called resmond to discover and monitor the device resources. Resmond polls the system resources to gather the status of the resource and store this data in the resource-specific BDS table.

The Resource Monitoring (resmon) functionality of RBFS provides support for monitoring the following components:

- CPU
- Memory
- Processes
- Disks
- Sensor
- Optics

CPU

Resmond collects CPU hardware information from the global.chassis_0.resource.cpu table. In addition, Resmond calculates CPU usage dynamically and stores this information in the global.chassis_0.resource.cpu_usage table.

Memory

Resmond collects RAM hardware information from the global.chassis_0.resource.mem table. In addition, Resmond gathers memory usage information in the global.chassis_0.resource.mem_usage table.

Processes

Resmond collects process usage information of Brick Daemon(BD) that runs in the RBFS and stores the information in the global.chassis_0.resource.proc_usage table. It dynamically gathers the process information and calculates the CPU and the memory usage of the individual Brick Daemons.

Disks

Resmond collects the disk information from the global.chassis_0.resource.disk table. In addition, Resmond collects disk usage information in the global.chassis_0.resource.disk_usage table.

Sensor

Resmond collects the reading data of the hardware sensor such as temperature, fan, power-supply, and system LED. The data collected from the sensor are stored in the global.chassis_0.resource.sensor table.

The RBFS implementation supports pluggable optics modules on white box switches only.

System Clock

Resmond provides support for monitoring the system clock so that the system clock is always in sync with the NTP server clock. This ensures that the deviation from the NTP server clock always remains within acceptable limits. Resmond collects the system clock information from the global.os.timex table.

The RBFS implementation supports pluggable optics modules on white box switches only.

Optical Modules

Resmond monitors optical transceivers plugged onto the chassis. It reads transceivers EEPROM (Electrically Erasable Programmable Read-Only Memory) data and translates the data to respective fields in the BDS tables.

Resmond provides the following functionalities for monitoring optical transceivers:

- Provides a mechanism to discover and monitor optics modules. Supported optics modules include SFP, SFP+, QSFP, QSFP+, and QSFP28 (DAC is not supported).
- Provides CLIs to write to optics modules
- Provides show commands to see optics inventory and status of each module
- Logs the status of the optics module

The RBFS implementation supports monitoring of pluggable optics modules on white box switches only.

The following are some of the important tasks (but not limited to) that the Resmond application performs:

- Optics inventory: Identifying the following brief information of a discovered optics module and stores in table global.chassis_0.resource.optics.inventory.

  Port

  Type

  Vendor

  Serial Number

  Part Number

- Read the following optics data from a module and stores in the table: global.chassis_0.resource.optics.module.

  RX/TX alarming (loss of light and loss of signal)

- RX/TX power status

  Voltage and BIAS status

  Temperature

- Write the optics data to an optics module

  Enabling high power class on QSFP28

  Shutdown lasers (QSFP28, SFP+ and SFP)

Optics Logging

The Resmond can log the following Optics module events:

- Temperature high alarm
- Temperature high warning
- Temperature low alarm
- Temperature low warning
- Voltage high alarm
- Voltage high warning

- Voltage low alarm
- Voltage low warning
- Lane power high alarm
- Lane power high warning
- Lane power low alarm
- Lane power low warning
- Lane bias high alarm
- Lane bias high warning
- Lane bias low alarm
- Lane bias low warning

Q2C Resource Monitoring

Q2C platform resource-specific usage metrics are stored in the BDS table: local.bcm.q2c.resource.monitor. Resource usage information enables you to understand the scale of services that the device performs and how it optimally leverages the resource usage.

The following table provides the list of supported resource types for monitoring in RBFS for Q2C platforms.

Resource Type	Description
EEDB_L2TP	EEDB is an Egress Encapsulation Data Base. This resource is consumed when L2TP subscribers are created in hardware.
EEDB_MPLS_TUNNEL	EEDB is an Egress Encapsulation Data Base. This resource is consumed when MPLS tunnels are created in the chip.
EEDB_PPPOE	EEDB_PPPOE is used for PPPoE encapsulation. This resource is consumed when PPPoE subscribers are created in hardware.
EEDB_PWE	This resource is consumed when L2X or cross-connection sessions are created in hardware at egress.

Resource Type	Description
IN_LIF_FORMAT_PWE	This resource is consumed when a pseudowire is created in the hardware at ingress.
IN_AC_C_C_VLAN_DB	This resource is consumed when an ingress logical interface for double-tagged VLAN is created.
IN_AC_C_VLAN_DB	This resource is consumed when an ingress logical interface for single tag VLAN is created.
IN_AC_UNTAGGED_DB	This resource is consumed when an ingress logical interface for untagged IFLs is created.
IPV4_MULTICAST_PRIVATE_LPM_FORWARD	LPM stands for Longest Prefix Match. This resource is consumed for multicast (source, group) entries.
IPV4_UNICAST_PRIVATE_LPM_FORWARD	This resource is consumed for non-default VRF instance IPv4 prefixes.
IPV4_UNICAST_PRIVATE_LPM_FORWARD_2	This resource is consumed for default VRF instance IPv4 prefixes.
IPV6_UNICAST_PRIVATE_LPM_FORWARD	This resource is consumed for non-default VRF instance IPv6 prefixes.
IPV6_UNICAST_PRIVATE_LPM_FORWARD_2	This resource is consumed for default VRF instance IPv6 prefixes.
L3_RIF	This resource is consumed for the L3 interfaces.
L2TPV2_DATA_MESSAGE_TT	This resource is consumed when the L2TP subscribers are created in hardware.
MPLS_FWD	This resource is consumed for MPLS entries for which forwarding actions are involved.
MPLS_TERMINATION_SINGLE_LABEL_DB	This resource is consumed for MPLS entries for which label termination is required.
MULTICAST_MCDB	This resource is consumed for multicast groups created in hardware.
PPPOE_O_ETH_TUNNEL_FULL_SA	This resource is consumed for PPPoE subscribers in hardware.

Example: Logical table information for the resource type EEDB_L2TP

```
supervisor@rtbrick: dbg> bcm "dbal table info table=EEDB_L2TP"
```

```
Logical table info  EEDB_L2TP
=============================

        Access method: MDB
        Table type: DIRECT
        Touched status: Initialized
        Entries Status: Max Capacity: HW dependent (see mapping), Committed 0
        Bulk mode range NOT supported
        Maturity_level: HIGH
        Table Labels: L2, L3, MPLS, EEDB
        Core mode: SBC
        Max key value: 1048575
        Max payload size in bits: 101
<...>
```

Example: Logical table information for the resource type EEDB_MPLS_TUNNELEEDB_MPLS_TUNNEL

```
supervisor@rtbrick: dbg> bcm "dbal table info table=EEDB_MPLS_TUNNEL"

Logical table info  EEDB_MPLS_TUNNEL
====================================

        Access method: MDB
        Table type: DIRECT
        Touched status: Initialized
        Entries Status: Max Capacity: HW dependent (see mapping), Committed 18
        Bulk mode range NOT supported
        Maturity_level: HIGH
        Table Labels: L2, L3, MPLS, EEDB
        Core mode: SBC
        Max key value: 1048575
        Max payload size in bits: 147
<...>
```

QAX Resource Monitoring

QAX platform resource-specific usage metrics are stored in the BDS table: local.bcm.qax.resource.monitor. Resource usage information enables you to understand the scale of services that the device performs and how it optimally leverages the resource usage.

The following table provides the list of supported resource types for monitoring in QAX platforms.

Resource Type	Description
ECMP ID	It represents an index in the table that maintains the paths for a particular ECMP route.

Resource Type	Description
Egress Failover ID	It identifies failover paths for traffic leaving the device.
FEC Failover ID	This is used to manage failover paths for FECs.
FECs for Global Use	It represents the number of FECs that are available for general use across the device.
Field Direct Extraction Entry ID	It represents the entries in the FDE table, which are used for direct extraction of fields.
Field Entry ID	It identifies a specific entry used for field-based operations.
Ingress Failover ID	It identifies backup paths for traffic entering the device.
Local Common InLif	It represents the number of common InLifs (Ingress Logical Interface) allocated for incoming traffic.
Local OutLif	It represents the number of logical interfaces allocated for outgoing traffic.
Local Wide InLif	Wide InLifs are used for additional metadata or special handling is required for incoming traffic. It represents the number of wide InLifs allocated for such purposes.
Number of Meters in Processor A	Meters are used for rate limiting or policing traffic, ensuring that traffic adheres to predefined bandwidth limits. This represents the number of meters available and used in processor A.
Profiles for PON Use	Profiles related to Passive Optical Networks are used in the management and configuration of PON interfaces. The number of profiles allocated for PON operations.
QOS EGRESS DSCP/EXP MARKING PROFILE IDs	These IDs represent the QoS marking profiles used to mark outgoing traffic based on DSCP or EXP bits, ensuring the prioritization of outgoing traffic.
QOS EGRESS L2 I TAG PROFILE IDs	These IDs are used for profiles that manage Layer 2 (L2) tagging in egress traffic. The number of available profiles for L2 I Tagging.

Resource Type	Description
QOS EGRESS MPLS PHP QOS IDs	The IDs identify QoS profiles used for MPLS PHP (Penultimate Hop Popping) operations.
QOS EGRESS REMARK QOS IDs	This is used to re-mark packets with a new QoS value as they exit the device. IDs represent the QoS profiles that apply new markings to egress traffic.
QOS INGRESS COS OPCODE IDs	COS (Class of Service) opcodes are used to classify incoming traffic based on criteria such as type of service. The IDs represent different COS opcode profiles for classifying ingress traffic.
QOS INGRESS LABEL MAP ID	Maps are used to associate incoming packets with certain labels for QoS processing. The number of available Label Maps for ingress traffic.
QOS INGRESS LIF/COS IDs	This ID is used for associating incoming traffic with specific LIF (Logical Interface) and COS profiles for handling. The number of IDs available for mapping ingress traffic to LIF/COS profiles.
QOS INGRESS PCP PROFILE IDs	Priority Code Point (PCP) profiles are used to prioritize packets at the ingress based on their PCP value. The IDs represent available PCP profiles for prioritizing ingress traffic.
SW Handles of Policer	It represents the number of software handles available for configuring policers.
Trill Virtual Nickname	It represents the number of virtual nicknames allocated for TRILL protocol operations for creating loop-free multi-path Ethernet network.
VLAN Translation Egress Usage	The number of entries used for VLAN translation operations on out-going traffic.
VLAN Translation Ingress Usage	The number of entries used for VLAN translation on incoming traffic.
VSIs for MSTP	The number of VSIs (Virtual Switch Instances) allocated for MSTP operations.
VSIs for TB VLANS	The number of VSIs allocated for handling TB VLANs.

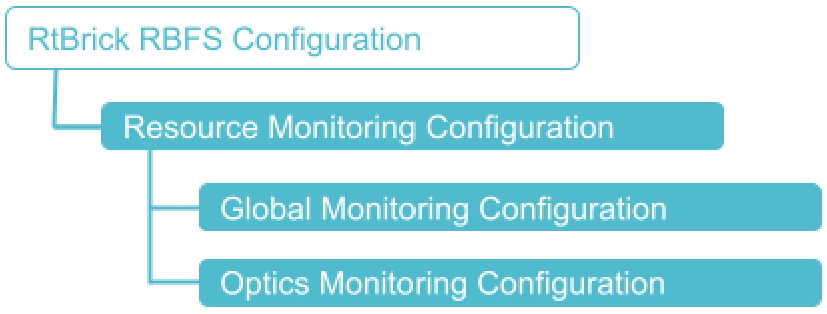
Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

16.2.2. Resmon Configuration

Configuration Hierarchy

The diagram illustrates the Resmon configuration hierarchy.


RtBrick RBFS Configuration
Resource Monitoring Configuration
Global Monitoring Configuration
Optics Monitoring Configuration

Configuration Syntax and Commands

The following sections describe the Resmon configuration syntax and commands.

Global Monitoring Configuration

This command sets the poll-interval for the resmond to discover optics.

Syntax:

set resmon monitoring <poll-interval>

Attribute	**Description**
poll_interval <poll-interval>	Specifies the interval (in seconds) at which optics should be polled. The interval can range from 3 to 10000 seconds. The default interval is 10 seconds.

Optics Configuration

You can use this command to disable or enable (By, default enabled) tx laser or high-power class of an optics module on a specific interface.

Syntax:

set resmon optics <interface> ...

Attribute	Description
interface <interface-name>	Name of the interface
high-power-class [disable / enable]	Enable or disable high power class for optics module. Enabled, by default.
tx [disable / enable]	Enable or disable lasers for optics module. Enabled, by default.

16.2.3. Resmon Operational Commands

Resmon Show Commands

The Resmon show commands provide detailed information about the resources and their usage.

CPU Information

This command displays the CPU detail and CPU information.

Syntax:

show cpu <option>

Option	Description
summary	Displays the CPU information.
usage	Displays the CPU usage information

Example 1: CPU summary

```
supervisor@rtbrick: op> show cpu summary
CPU_0
  Vendor                : GenuineIntel
  Model                 : Intel(R) Xeon(R) CPU D-1518 @ 2.20GHz
  Architecture          : x86_64
```

```
    Serial No            : 12 34 56 78 AB CD EF GH
    Clock(MHz)           : 1996.620
    BogoMIPS             : 4400.00
    Physical cores       : 4
    Logical cores        : 8
    Endian               : True
    Cache alignment      : 64 Bytes
    L1 data cache        : 32768 Bytes
    L1 instruction cache : 32768 Bytes
    L2 unified cache     : 262144 Bytes
    L3 unified cache     : 6291456 Bytes
    L4 unified cache     : 0 Bytes
supervisor@rtbrick: op>
```

Example 2: CPU usage information.

```
supervisor@rtbrick: op> show cpu usage
Name         Total       User     System       Nice   I/O Wait       Idle        IRQ   Soft IRQ
cpu            31%        23%         8%         0%         0%        68%         0%         0%
cpu0           11%         4%         5%         0%         0%        88%         0%         2%
cpu1          100%        63%        36%         0%         0%         0%         0%         0%
cpu2           54%        51%         2%         0%         0%        45%         0%         0%
cpu3           57%        55%         2%         0%         0%        42%         0%         0%
cpu4            6%         1%         5%         0%         0%        93%         0%         0%
cpu5            4%         2%         2%         0%         0%        95%         0%         0%
cpu6            8%         3%         4%         0%         0%        91%         0%         0%
cpu7            6%         1%         5%         0%         0%        94%         0%         0%
supervisor@rtbrick: op>
```

Memory Details

This command displays the memory details and usage information.

Syntax:

show memory <option>

Option	**Description**
summary	Displays the system memory information.
usage	Displays the memory usage information.

Example 1: System memory information

```
supervisor@rtbrick: op> show memory summary
System Memory
  Maximum capacity                : 128 GB
  Error correction type           : Multi-bit ECC
  Number of memory slots available : 4
  Number of memory slots occupied  : 2
Bank              Size        Location         Type      Speed       Configured Speed    Vendor      Serial
No      Part No
NODE 1            8192 MB     DIMM_A1          DDR4      2133 MT/s   2133 MT/s           Undefined   00000001
```

```
TS1GSH72V1H
NODE 1          8192 MB     DIMM_B1           DDR4      2133 MT/s   2133 MT/s          Undefined   00000002
TS1GSH72V1H
```

Example 2: System memory usage information.

```
supervisor@rtbrick: op> show memory usage
Name      Total             Used              Free         Shared       Buffers      Cached
RAM       16.69 GB          4.54 GB           10.08 GB     578.17 MB    103.12 MB    2.06 GB
SWAP      0 bytes           0 bytes           0 bytes      n/a          n/a          n/a
```

Process Details

This command displays the process usage information.

Syntax:

show process usage <option>

Option	Description
process-id <pid>	Displays the process usage for the specified process identifier.
process-name <process-name>	Displays the process usage for the specified process.
summary	Displays the process usage summary information.

Example 1: Process usage summary information.

```
supervisor@rtbrick: op> show process usage summary
Name             PID         VIRT              Resident Memory      Sharable Memory     CPU Percentage
Memory Percentage   CPU Affinity
bgp.appd.1       4456        384.67 MB         122.69 MB            29.32 MB            0.81
0.74                0-7
bgp.iod.1        4469        694.63 MB         148.39 MB            30.22 MB            2.01
0.89                0-7
confd            213         1.24 GB           769.9 MB             31.89 MB            0.81
4.61                0-7
etcd             110         600.9 MB          196.4 MB             29.26 MB            1.21
1.18                0-7
fibd             288         13.68 GB          1.74 GB              210.79 MB           161.17
10.45               1
ifmd             147         496.74 MB         154.25 MB            29.99 MB            1.01
0.92                0-7
igmp.appd.1      4482        356.38 MB         100.29 MB            29.27 MB            1.01              0.6
0-7
igmp.iod.1       4495        540.14 MB         132.68 MB            29.41 MB            2.01              0.8
0-7
ipoed.1          175         503.51 MB         112.9 MB             29.3 MB             2.01
0.68                0-7
l2tpd.1          4508        475.71 MB         103.45 MB            29.26 MB            2.01
0.62                0-7
lldpd            128         368.32 MB         106.75 MB            29.29 MB            2.21
0.64                0-7
```

```
mribd            158        381.8 MB         115.57 MB           29.38 MB           1.01
0.69                0-7
```

Example 2: Process usage for the specified process.

```
supervisor@rtbrick: op> show process usage process-name fibd
Process Name: fibd
  PID                          : 288
  REST port                    : 5522
  Debug port                   : 5521
  Allowed CPU list             : 1
  CPU usage at user space      : 25964
  CPU usage at kernel space    : 12633
  CPU usage percentage         : 152.815678
  Memory usage percentage      : 10.452729
  Peak virtual memory usage    : 13355692
  Current virtual memory usage : 13.68 GB
  <...>
```

Example 3: Process usage for the specified process ID.

```
supervisor@rtbrick: op> show process usage process-id 4456
Process Name: bgp.appd.1
  PID                          : 4456
  REST port                    : 4102
  Debug port                   : 4101
  Allowed CPU list             : 0-7
  CPU usage at user space      : 20103
  CPU usage at kernel space    : 18011
  CPU usage percentage         : 1.011122
  Memory usage percentage      : 0.735218
  Peak virtual memory usage    : 375652
  Current virtual memory usage : 384.67 MB
  Locked virtual memory        : n/a
  RSS virtual memory           : 122.69 MB
  <...>
```

Sensor Details

This command displays the fan, power-supply, system-led, and temperature information.

Syntax:

show sensor <option>

Option	Description
fan	Displays information about the sensor fan.

Option	Description
power-supply	Displays the sensor power supply information.
system-led	Displays system LED information
temperature	Displays the sensor temperature information.
detail	You can specify detail at the end of any of the options above to display information in detail.

Example 1: Sensor temperature information.

```
supervisor@rtbrick: op> show sensor temperature
Name                     Temperature         Status
CPU Core                 54 °C               PRESENT
LM75-1-48                35 °C               PRESENT
LM75-2-49                30 °C               PRESENT
LM75-3-4A                33 °C               PRESENT
LM75-3-4B                30 °C               PRESENT
PSU-1 Thermal Sensor 1   31 °C               PRESENT
supervisor@rtbrick: op>
```

Example 2: Detailed information about the sensor temperature.

```
supervisor@rtbrick: op> show sensor temperature
Name                     Temperature         Status
CPU Core                 54 °C               PRESENT
LM75-1-48                36 °C               PRESENT
LM75-2-49                30 °C               PRESENT
LM75-3-4A                33 °C               PRESENT
LM75-3-4B                30 °C               PRESENT
PSU-1 Thermal Sensor 1   31 °C               PRESENT
supervisor@rtbrick: op>
```

Example 3: Information about sensor fan

```
supervisor@rtbrick: op> show sensor fan
Name               Fan Speed (rpm)     Status
PSU 1 - Fan 1      26496               PRESENT, F2B
Chassis Fan - 1    8700                PRESENT, F2B
Chassis Fan - 2    8700                PRESENT, F2B
Chassis Fan - 3    8700                PRESENT, F2B
Chassis Fan - 4    8700                PRESENT, F2B
Chassis Fan - 5    8700                PRESENT, F2B
Chassis Fan - 6    8600                PRESENT, F2B
supervisor@rtbrick: op>
```

Example 4: Detailed information about sensor fan.

```
supervisor@rtbrick: op> show sensor fan detail

PSU 1 - Fan 1
  Sensor resource ID   : 8388614
  Vendor               : n/a
  Model                : NULL
  Serial No            : n/a
  Status               : PRESENT, F2B
  Status code          : 9
  Fan speed            : 26496 rpm
  Location             : PSU 1
Chassis Fan - 1
  Sensor resource ID   : 8388608
  Vendor               : ALTERA
  Model                : 5M1270ZF256C5N
  Serial No            : n/a
  Status               : PRESENT, F2B
  Status code          : 9
  Fan speed            : 8700 rpm
  Location             : Fan Board
Chassis Fan - 2
  Sensor resource ID   : 8388609
  Vendor               : ALTERA
  Model                : 5M1270ZF256C5N
  Serial No            : n/a
  Status               : PRESENT, F2B
  Status code          : 9
  Fan speed            : 8800 rpm
  Location             : Fan Board
```

Example 5: Sensor power supply information

```
supervisor@rtbrick: op> show sensor power-supply
Name              Current In  Current Out Voltage In  Voltage Out Power In    Power Out   Status
PSU-1             0 mA        12109 mA    0 mV        11984 mV    0 mW        146000 mW   PRESENT
PSU-2             0 mA        0 mA        0 mV        0 mV        0 mW        0 mW        PRESENT, UNPLUGGED
supervisor@rtbrick: op>
```

Example 6: Detailed information about the sensor power supply.

```
supervisor@rtbrick: op> show sensor power-supply detail

PSU-1
  Sensor resource ID   : 16777216
  Vendor               : n/a
  Model                : YM-2651Y
  Serial No            : n/a
  Status               : PRESENT
  Status code          : 1
  Input current        : 0 mA
  Output current       : 12031 mA
  Input voltage        : 0 mV
  Output voltage       : 11984 mV
  Input power          : 0 mW
  Output power         : 144000 mW
  Location             : n/a
```

```
PSU-2
  Sensor resource ID   : 16777217
  Vendor               : n/a
  Model                : NULL
  Serial No            : n/a
  Status               : PRESENT, UNPLUGGED
  Status code          : 5
  Input current        : 0 mA
  Output current       : 0 mA
  Input voltage        : 0 mV
  Output voltage       : 0 mV
  Input power          : 0 mW
  Output power         : 0 mW
  Location             : n/a
supervisor@rtbrick: op>
```

Example 7: Sensor system LED information

```
supervisor@rtbrick: op> show sensor system-led
Name                      LED Mode         Status
Chassis LED 1 (LOC LED)  OFF              PRESENT
Chassis LED 5 (FAN LED)  AUTO             PRESENT, ON
Chassis LED 2 (DIAG LED) OFF              PRESENT
Chassis LED 3 (PSU1 LED) AUTO             PRESENT, ON
Chassis LED 4 (PSU2 LED) AUTO             PRESENT, ON
supervisor@rtbrick: op>
```

Example 8: Detailed information about the system LED

```
supervisor@rtbrick: op> show sensor system-led detail

Chassis LED 1 (LOC LED)
  Sensor resource ID   : 12582912
  LED mode             : OFF
  Status               : PRESENT
  Status code          : 1
  Capability           : ON_OFF, ORANGE
  Capability code      : 4097
Chassis LED 5 (FAN LED)
  Sensor resource ID   : 12582916
  LED mode             : AUTO
  Status               : PRESENT, ON
  Status code          : 5
  Capability           : ON_OFF, AUTO
  Capability code      : 4194305
Chassis LED 2 (DIAG LED)
  Sensor resource ID   : 12582913
  LED mode             : OFF
  Status               : PRESENT
  Status code          : 1
  Capability           : ON_OFF, ORANGE, GREEN
  Capability code      : 69633
```

Optics Details

This command displays optics information for inventory and interface.

Syntax:

show optics <option>

Option	Description
interface <interface-name>	Displays optics information for the specified interface.
inventory	Displays optics inventory information.

Example: Optics information for the specified interface.

```
supervisor@rtbrick: op> show optics interface ifp-0/1/6
Physical Interface: ifp-0/1/6
Type                   : QSFP28
Description            : 100G-CWDM4
Connector Type         : Lucent Connector
Vendor                 : FS
Serial Number          : E1234567890
Part Number            : QSFP28-IR4-100G
Vendor Material Number : ABCDEFGHIJ12-3456-78
Power Class            : Class 4 (3.5W)
Power Class State      : HIGH
Wavelength             : 1310.000000
Lane Id                : 1
  Laser bias current                         : 30.824 mA
  Laser tx power                             : 1.496 mW / 1.749 dbm
  Laser rx power                             : 1.084 mW / 0.35 dbm
  Module temperature                         : 39.164 °C
  Module voltage                             : 3.292 V
  Tx disable                                 : False
  High power class enable                    : True
  Laser Tx loss of signal                    : False
  Laser Tx loss of lock                      : False
  Laser Rx loss of signal                    : False
  Laser Rx loss of lock                      : False
  Laser bias current high alarm              : False
  Laser bias current high warning            : False
  Laser bias current low alarm               : False
  Laser bias current low warning             : False
  Module voltage high alarm                  : False
  Module voltage high warning                : False
  <...>
```

Example 2: Optics inventory information.

```
supervisor@rtbrick: op> show optics inventory
```

```
Interface      Type     Description    Connector Type          Vendor        Part Number         Serial
Number    Material Number    Power Class      Power State
ifp-0/1/2      QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC005
A1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/3      QSFP28   100G-CWDM4     Lucent Connector        LambdaGain    LL1C31A2A           B123456789
A-ABCEFG12345678  Class 4 (3.5W)   HIGH
ifp-0/1/4      QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC01
C1234567890-2   n/a               Class 1 (1.5W)   LOW
ifp-0/1/5      QSFP28   100GBASE-LR4   Lucent Connector        LambdaGain    LL1S31B0A           D123456789
A-ABCXYZ12345678  Class 4 (3.5W)   HIGH
ifp-0/1/6      QSFP28   100G-CWDM4     Lucent Connector        FS            QSFP28-IR4-100G
E1234567890     ABCDEFGHIJ12-345  Class 4 (3.5W)   HIGH
ifp-0/1/8      QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC005
F1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/9      QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC005
G1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/11     QSFP28   100GBASE-CR4   No Seperable connector  Fiberstore    QSFP28-100G-DAC
H1234567890-2   n/a               Class 1 (1.5W)   LOW
ifp-0/1/14     QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC005
I1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/15     QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC01
J1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/18     QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC01
K1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/20     QSFP28   100G-CWDM4     Lucent Connector        LambdaGain    LL1C31A2A           L123456789
A-ABCEFG12345678  Class 4 (3.5W)   HIGH
ifp-0/1/27     QSFP28   100GBASE-LR4   Lucent Connector        LambdaGain    LL1S31B0A           M123456789
A-ABCXYZ12345678  Class 4 (3.5W)   HIGH
ifp-0/1/28     QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC01
N1234567890-1   n/a               Class 1 (1.5W)   LOW
ifp-0/1/31     QSFP28   100GBASE-CR4   No Seperable connector  FS            Q28-PC01
O123456789-2   n/a               Class 1 (1.5W)   LOW
ifp-0/0/0      SFP      UNKNOWN        Copper Pigtail          FS            SFPP-PC015
P1234567890-2   n/a               Class 1 (1.5W)   LOW
ifp-0/0/1      SFP      UNKNOWN        Copper Pigtail          FS            SFPP-PC015
Q1234567890-2   n/a               Class 1 (1.5W)   LOW
```

To access the Operational State API that corresponds to this CLI, click here.

Disk Details

This command displays disk information.

Syntax:

show disk <option>

Option	Description
summary	Displays the disk information.
usage	Displays the disk usage information.

Example: Summary of disks and their partitions.

```
supervisor@rtbrick: op> show disk summary
sda
```

```
  Size    : 29.8G
  Vendor  : ATA
  Model   : TS32ZBTMM1600
  Partitions
  Name               Size        Mountpoint
  sda1               256M        n/a
  sda2               128M        n/a
  sda3               2G          n/a
  sda4               128M        n/a
  sda5               128M        n/a
  sda6               2G          n/a
  sda7               25.2G       /platform
supervisor@rtbrick: op>
```

Example 2: Summary of disk usage information.

```
supervisor@rtbrick: op> show disk usage
Filesystem                Type             Size         Used         Available    Mountpoint          Usage %
none                      tmpfs            492 KB       0 bytes      492 KB       /dev                0.0
tmpfs                     tmpfs            8.15 GB      17.38 MB     8.13 GB      /run                0.21
tmpfs                     tmpfs            6.29 GB      483.6 MB     5.81 GB      /shm                7.69
tmpfs                     tmpfs            8.15 GB      62.74 MB     8.09 GB      /dev/shm            0.77
tmpfs                     tmpfs            5.12 MB      0 bytes      5.12 MB      /run/lock           0.0
devtmpfs                  devtmpfs         1.02 MB      0 bytes      1.02 MB      /dev/mem            0.0
/dev/sda7                 ext4             25.87 GB     4.47 GB      20.06 GB     /var/log            18.21
tmpfs                     tmpfs            1.63 GB      0 bytes      1.63 GB      /run/user/1000      0.0
tmpfs                     tmpfs            8.15 GB      0 bytes      8.15 GB      /sys/fs/cgroup      0.0
tmpfs                     tmpfs            1.63 GB      696 KB       1.63 GB      /var/run-ext/onl/r  0.04
/var/cache/rtbrick/imag   overlay          25.87 GB     4.47 GB      20.06 GB     /                   18.21
supervisor@rtbrick: op>
```

Platform Details

This command displays platform information.

Syntax:

show platform

Example: Platform information

```
supervisor@rtbrick: op> show platform
x86_64-accton_as7316_26xb-r0
  Vendor                : Edgecore
  Manufacturer          : Accton
  Manufacture date      : 05/07/2021 16:55:51
  MAC address           : 90:3c:b3:16:00:00
  Part number           : F0PZZ5626002A-MACDDR-Nanya_NT5AD256M16D4_HRI
  Serial number         : ABC1234ABCD
  Product name          : 7316-26XB-O-AC-F
  Onie version          : 2019.11.00.07
  Label revision        : R01C
  Diag version          : 01.01.00.03
  Country code          : TW
supervisor@rtbrick: op>
```

Hardware Resource Usage Information

The command show hardware resource monitor displays the hardware consumption information for the various resources.

Syntax:

show hardware resource monitor

Example: Hardware resource monitoring information

```
supervisor@rtbrick: op> show hardware resource monitor
   Resource Type                                          Consumed Estimated
   EEDB_L2TP                                                     0 49152
   EEDB_MPLS_TUNNEL                                              0 49152
   EEDB_PPPOE                                                    0 49152
   EEDB_PWE                                                      0 49152
   IN_AC_C_C_VLAN_DB                                             0 60292
   IN_AC_C_VLAN_DB                                              36 60292
   IN_AC_UNTAGGED_DB                                             3 60292
   IPV4_MULTICAST_PRIVATE_LPM_FORWARD                            0 568320
   IPV4_UNICAST_PRIVATE_LPM_FORWARD                             11 1136640
   IPV4_UNICAST_PRIVATE_LPM_FORWARD_2                           11 378880
   IPV6_UNICAST_PRIVATE_LPM_FORWARD                              7 568320
   IPV6_UNICAST_PRIVATE_LPM_FORWARD_2                           11 189440
   KBP_IPV4_UNICAST_PRIVATE_LPM_FORWARD                          0 0
   KBP_IPV6_UNICAST_PRIVATE_LPM_FORWARD                          0 0
   L2TPV2_DATA_MESSAGE_TT                                        0 60292
   L3_RIF                                                       51 40959
   MPLS_FWD                                                      0 1017112
   MPLS_TERMINATION_SINGLE_LABEL_DB                             14 60292
   MULTICAST_MCDB                                                2 262143
   PPPOE_O_ETH_TUNNEL_FULL_SA                                    0 24576
```

16.3. SNMP

16.3.1. SNMP Overview

SNMP (Simple Network Management Protocol) provides a network monitoring mechanism that collects state information from various network devices and components. The protocol enables you to monitor the RBFS network and detect network faults on remote devices.

SNMP can monitor interfaces, CPU usage, temperature of the device, bandwidth usage, and so on. For example, if an interface goes down on one of the devices, SNMP can quickly alert the change.

Understanding RBFS SNMP Implementation

The RBFS SNMP implementation allows retrieving system state information using the Protocol Data Unit (PDU) from various network components. SNMP defines system objects in Management Information Bases (MIBs), where each object forms a Protocol Data Unit (PDU).

A device that is SNMP enabled is known as SNMP agent. SNMP agent collects information from various devices and components and stores the data within MIB. An SNMP agent includes several objects such as interfaces and routing tables which can be interacted with. Every object has a unique identifier which is known as OID (Object Identifier). OIDs provide a unique identity for managed objects in an MIB hierarchy.

RBFS implements the SNMP daemon (snmpd) that maps the operational state API to SNMP MIBs for retrieving system state data.

SNMP Operations

SNMP allows performing various operations that include GET for retrieving data, SET for modifying data, TRAP for notifying an event and so on. These operations provide management access to the MIB hierarchy.

1. **GET**: The SNMP GET operation retrieves data from the managed device’s MIB.
2. **GETNEXT**: The GETNEXT operation retrieves data for the next object from the tree of objects on the device.
3. **SET**: The SNMP SET operation allows modifying data for a device.
4. **TRAP**: Event notification that is not requested.

RBFS does not currently support the SNMP SET operation.

Supported SNMP MIBs

Management Information Base (MIB) is a collection of data that is organized hierarchically. You can define an MIB mapping for the supported MIBs. This MIB mapping provides a way to retrieve the required state information using the Operational State API or any other RBFS API or Prometheus.

Currently, RBFS supports two SNMP MIBs: the Interface MIB and the Host Resources MIBs.

- **Host Resources MIB**: Provides information about system resources on the host machine, such as CPU, memory, and storage. For details, see https://www.net-snmp.org/docs/mibs/HOST-RESOURCES-MIB.txt.
- **Interfaces MIB**: Provides information about the status and statistics of network interfaces. For details, see https://www.net-snmp.org/docs/mibs/IF-MIB.txt.
- **System MIB**: Supplies general device information, including system name, hardware details, and the installed network operating system version. https://www.net-snmp.org/docs/mibs/RFC1213-MIB.txt.

Supported SNMP Versions

RBFS supports the SNMP version 2c and SNMP version 3. SNMP version 3 provides support for authentication and encryption and the data can be accessed after capture only by authorized users. You must choose either version 2c or version 3 before configuring other functionalities.

Information about SNMP v2c and v3

SNMP v2c allows access control through Communities, but the community information is not protected. Anyone with the community name can access all the information available for that community.

SNMP v3 provides a higher level of security using authentication and privacy protocols. RBFS supports the user-based security model defined in RFC 3414.

The user credentials verify the authenticity and integrity of the message by adding message authentication codes (MACs) to the SNMP packet. The privacy protocol allows for the encryption of the transmitted data.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

16.3.2. SNMP Configuration

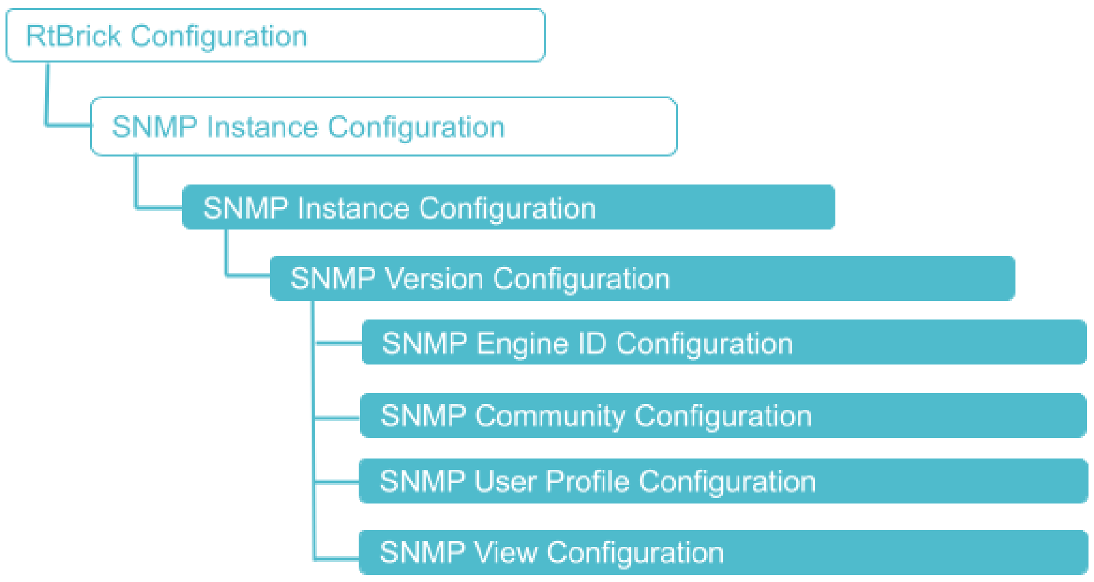
By default, SNMP is not enabled. To enable SNMP, you must complete the SNMP configurations.

RBFS supports SNMP 2c and 3 versions. You must first configure the desired version before configuring other functionalities. RBFS does not support running both the SNMP 2c and 3 versions at the same time.

The RBFS CLI displays all options regardless of the selected version. Even though functionalities such as Community can be configured for both of the versions, it works only in SNMP 2c version. You can define 'user profiles' on SNMP v3 and 'Community' in version 2c. Similarly, Engine ID can be defined in SNMP version 3.

Configuration Hierarchy

The diagram illustrates the SNMP configuration hierarchy. All SNMP configurations are performed within an instance.


RtBrick Configuration
SNMP Instance Configuration
SNMP Instance Configuration
SNMP Version Configuration
SNMP Engine ID Configuration
SNMP Community Configuration
SNMP User Profile Configuration
SNMP View Configuration

Configuration Syntax and Commands

The following sections describe the SNMP configuration syntax and commands.

SNMP Instance Configuration

At this instance configuration hierarchy, you configure SNMP protocol parameters which are generic to the SNMP instance.

Syntax

set instance <instance-name> **protocol snmp** <attribute> <value>

Attribute	Description
version	Specify the SNMP version. RBFS supports SNMP version 2c and version 3. You must first configure the desired version before configuring other functionalities.
engine-id	Specify the unique SNMP engine identifier. This is optional. If not specified, the system retrieves the default engine ID from the management port MAC address. Every SNMP v3 agent includes an engine ID that is a unique identifier for the agent. The engine ID is used to provide a higher level of security using authentication and encryption for SNMP v3 messages.

Example: SNMP Version and Engine Identifier Configuration

The following commands configure SNMP version 3 and engine ID: 268956.

```
set instance default protocol snmp version 3
set instance default protocol snmp engine-id 268956
```

The following example shows the SNMP version and engine ID configurations.

```
supervisor@rtbrick.net: cfg> show config instance default protocol snmp
{
  "rtbrick-config:snmp": {
    "version": "3",
    "engine-id": "268956"
  }
}
```

SNMP Community Configuration

An SNMP community can be defined only in the SNMP version 2c.

Syntax

set instance <instance-name> **protocol snmp community**

Attribute	Description
access-mode	Specify the access mode. Read, write and append are modes of access. 'ReadOnly' is the currently supported access mode.
view	Specify the list of view identifiers. View is optional. For information about Views, see section "2.2.3 SNMP View Configuration".

Example: SNMP v2c Community Configuration

The following commands configure a Community named 'public' with read-only access right to the 'interfaces' View.

```
set instance default protocol snmp version 2c
set instance default protocol snmp community public access-mode ReadOnly
set instance default protocol snmp community public view Interfaces
```

The following example shows SNMP v2c community configurations.

```
supervisor@rtbrick: cfg> show config instance default protocol snmp
{
  "rtbrick-config:snmp": {
    "version": "2c",
    "community": [
      {
        "name": "public"
        "access-mode": "ReadOnly"
      }
    ]
  }
}
```

SNMP View Configuration

An SNMP View is a subset of MIB objects. Views allow you to restrict access to certain items in the SNMP PDUs. You can restrict user and community access to certain attributes by defining views. A view restricts access to the PDUs included in the View. If the access is not restricted by views, the user or community is allowed to view all data available through SNMP.

Syntax

set instance <instance-name> **protocol snmp** <attribute> <value>

Attribute	Description
include <include>	List of OID patterns that are included in the view.
instance	List of instances. It restricts the view to the specified instances. If no instance is defined, the view can access to all instances.

Example: SNMP View Configuration

The following commands configure SNMP View. In this example configuration, SNMP version has been specified as 2c and 'View' name is specified as interfaces. The 'interfaces' view includes the OID 1.3.6.1.2.1.2.* in the view list. In addition, the configuration shows a user 'community' named public has been configured and the community has read-only access to the View.

```
set instance default protocol snmp version 2c
set instance default protocol snmp view interfaces include 1.3.6.1.2.1.2.*
set instance default protocol snmp community public access-mode ReadOnly
```

The following example shows the SNMP View configuration.

```
supervisor@rtbrick: cfg> show config instance default protocol snmp
{
  "rtbrick-config:snmp": {
    "version": "2c",
    "view": [
      {
        "name": "interfaces",
        "include": [
          "1.3.6.1.2.1.2.*"
          ]
      }
    ],
    "community": [
      {
        "name": "public",
        "access-mode": "ReadOnly",
        "view": [
          "interfaces"
          ]
      }
    ]
  }
}
```

SNMP Trap Configuration

The trap is a notification about a specific condition or event that occurs on the device. RBFS supports event notifications for events: link up and link down. Unlike other PDU types, a trap is a notification that is sent without any request from the SNMP manager.

Syntax

set instance <instance-name> **protocol snmp trap** <trap-name> <attribute> <value>

Attribute	Description
community	Community is supported in SNMPv2c only. Specify the 'Community' to enable trap notification for SNMPv2c.
include	List of OIDs that are included for the trap notifications.
instance	Specify the instance.
trap-receiver	Specify trap receiver device.
user-profile	User profile is supported on SNMPv3 only. Specify the 'user profile' to enable trap notification for SNMPv3.

SNMP User Profile Configuration

You can create user profiles for SNMP version 3. It allows you to define login credentials, authentication methods, and privacy control.

Syntax

set instance <instance-name> **protocol snmp user-profile**

Attribute	Description
authentication-protocol	Specify SNMP authentication protocol. MD5, NoAuth, SHA, SHA224, SHA256, SHA384, and SHA512 are the supported authentication protocol.
password-encrypted-text	Specify SNMP user password in encrypted text.
password-plain-text	Specify SNMP user password in plain text.

Attribute	Description
privacy-password-encrypted-text	Specify SNMP privacy password in encrypted text.
privacy-password-plain-text	Specify SNMP privacy password in plain text.
privacy-protocol	Specify SNMP privacy protocol. Supported privacy protocols include AES192, AES192C, AES256, AES256C, DES, and NoPriv.
security-level	Specify SNMP v3 security level. Security levels exist only in SNMP v3. The following security levels are supported: • noAuthNoPriv: no authentication, no privacy • authNoPriv: authentication, no privacy • authPriv: authentication, privacy
view	Specify SNMP view list.

Example: SNMP User Profile Configuration

The following commands configure SNMP user profile. At first, SNMP Version 3 is configured with the user profile name as operator. Password type has been selected as password encrypted text. In this configuration, the security level is configured as AuthNoPriv and MD5 as type of the authentication protocol.

```
set instance default protocol snmp version 3
set instance default protocol snmp user-profile operator
set instance default protocol snmp user-profile operator password-encrypted-text
$2a6fd7db50a18a9f1f16b5c5b4214fab0
set instance default protocol snmp user-profile operator security-level AuthNoPriv
set instance default protocol snmp user-profile operator authentication-protocol
MD5
```

The following example shows the SNMP User Profile Configuration

```
supervisor@rtbrick: cfg> show config instance default protocol snmp
{
  "rtbrick-config:snmp": {
    "version": "3",
    "user-profile": [
      {
        "name": "operator",
        "password-encrypted-text": "$2a6fd7db50a18a9f1f16b5c5b4214fab0",
        "security-level": "AuthNoPriv",
        "authentication-protocol": "MD5"
      }
    ]
  }
}
```

SNMP Location Configuration

With the Location option, you can configure basic administrative information such as the physical location for a managed device.

Syntax

set instance <instance-name> **protocol snmp location** <location-name>

Example Command:

```
set instance default protocol snmp location BLR
```

SNMP Contact Configuration

With the SNMP Contact option, you can configure basic administrative information for a managed device. You can specify the contact information of the person responsible for the device.

Syntax

set instance <instance-name> **protocol snmp contact** <contact-name>

Example Command:

```
set instance default protocol snmp contact RtBrick
```

Examples for SNMP Walk Operation

SNMP v2c SNMP Walk Output

The following is a sample output for the SNMP Walk for SNMP version 2c. SNMP version 2c has been configured with Community name as 'public' and, host IP address as 10.200.134.25.

```
snmpwalk -v 2c -c public 10.200.134.25
iso.3.6.1.2.1.2.2.1.1.1 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.3 = INTEGER: 3
iso.3.6.1.2.1.2.2.1.1.4 = INTEGER: 4
iso.3.6.1.2.1.2.2.1.1.5 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.1.6 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.1.7 = INTEGER: 7
iso.3.6.1.2.1.2.2.1.1.8 = INTEGER: 8
iso.3.6.1.2.1.2.2.1.1.9 = INTEGER: 9
iso.3.6.1.2.1.2.2.1.1.10 = INTEGER: 10
iso.3.6.1.2.1.2.2.1.1.11 = INTEGER: 11
iso.3.6.1.2.1.2.2.1.1.12 = INTEGER: 12
iso.3.6.1.2.1.2.2.1.1.13 = INTEGER: 13
iso.3.6.1.2.1.2.2.1.1.14 = INTEGER: 14
iso.3.6.1.2.1.2.2.1.1.15 = INTEGER: 15
iso.3.6.1.2.1.2.2.1.1.16 = INTEGER: 16
iso.3.6.1.2.1.2.2.1.1.17 = INTEGER: 17
iso.3.6.1.2.1.2.2.1.1.18 = INTEGER: 18
iso.3.6.1.2.1.2.2.1.1.19 = INTEGER: 19
iso.3.6.1.2.1.2.2.1.1.20 = INTEGER: 20
iso.3.6.1.2.1.2.2.1.1.21 = INTEGER: 21
iso.3.6.1.2.1.2.2.1.1.22 = INTEGER: 22
iso.3.6.1.2.1.2.2.1.1.23 = INTEGER: 23
<...>
```

SNMP v3 SNMP Walk Output

The following is a sample output for the SNMP Walk for SNMP version 3. SNMP version 3 has been configured with user as 'operator', MD5 as the authentication protocol, authNoPriv as the security level, and 10.200.134.25 as the host IP address.

```
snmpwalk -v 3 -u operator -A operator -a MD5 -l authNoPriv 10.200.134.25
iso.3.6.1.2.1.2.2.1.1.1 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.3 = INTEGER: 3
iso.3.6.1.2.1.2.2.1.1.4 = INTEGER: 4
iso.3.6.1.2.1.2.2.1.1.5 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.1.6 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.1.7 = INTEGER: 7
iso.3.6.1.2.1.2.2.1.1.8 = INTEGER: 8
iso.3.6.1.2.1.2.2.1.1.9 = INTEGER: 9
iso.3.6.1.2.1.2.2.1.1.10 = INTEGER: 10
iso.3.6.1.2.1.2.2.1.1.11 = INTEGER: 11
iso.3.6.1.2.1.2.2.1.1.12 = INTEGER: 12
iso.3.6.1.2.1.2.2.1.1.13 = INTEGER: 13
iso.3.6.1.2.1.2.2.1.1.14 = INTEGER: 14
iso.3.6.1.2.1.2.2.1.1.15 = INTEGER: 15
iso.3.6.1.2.1.2.2.1.1.16 = INTEGER: 16
iso.3.6.1.2.1.2.2.1.1.17 = INTEGER: 17
iso.3.6.1.2.1.2.2.1.1.18 = INTEGER: 18
iso.3.6.1.2.1.2.2.1.1.19 = INTEGER: 19
iso.3.6.1.2.1.2.2.1.1.20 = INTEGER: 20
iso.3.6.1.2.1.2.2.1.1.21 = INTEGER: 21
iso.3.6.1.2.1.2.2.1.1.22 = INTEGER: 22
iso.3.6.1.2.1.2.2.1.1.23 = INTEGER: 23
<...>
```

The following is a sample output for the SNMP System MIB walk operation.

```
snmpwalk -v 2c -c ufi17.q2c.u34.r6.nbg.rtbrick.net"
<...>
iso.3.6.1.2.1.1.1 = STRING: "UfiSpace S9600-102XC-R, RBFS 24.9.0-
g6daily.20241007154425+Bdevelopment.C8e6a3a56"
iso.3.6.1.2.1.1.2 = OID: iso.3.6.1.4.1.50058.1.1
iso.3.6.1.2.1.1.3 = Timeticks: (35079) 0:05:50.79
iso.3.6.1.2.1.1.4 = ""
iso.3.6.1.2.1.1.5 = STRING: "ufi17.q2c.u34.r6.nbg.rtbrick.net"
iso.3.6.1.2.1.1.6 = ""
iso.3.6.1.2.1.1.7 = INTEGER: 12
iso.3.6.1.2.1.1.7 = No more variables left in this MIB View (It is past the end
of the MIB tree)
<...>
```

Example 2:

```
snmpwalk -v 2c -c rbfs ufi11.q2a.u15.r6.nbg.rtbrick.net
iso.3.6.1.2.1.1.1 = STRING: "Ufi Space S9510-28DC-B, RBFS 25.1.0-
g6daily.20250415154806+Bdevelopment.Cdf3d9037"
iso.3.6.1.2.1.1.2 = OID: iso.3.6.1.4.1.50058.1.1
iso.3.6.1.2.1.1.3 = Timeticks: (41382) 0:06:53.82
iso.3.6.1.2.1.1.4 = STRING: "RtBrick"
iso.3.6.1.2.1.1.5 = STRING: "ufi11.q2a.u15.r6.nbg.rtbrick.net"
iso.3.6.1.2.1.1.6 = STRING: "BLR"
iso.3.6.1.2.1.1.7 = INTEGER: 12
iso.3.6.1.2.1.1.7 = No more variables left in this MIB View (It is past the end of
the MIB tree)
```

17. Infrastructure

17.1. NTP

17.1.1. NTP Overview

Network Time Protocol (NTP) provides accurate time across a whole network of devices such as routers or switches and synchronizes time among these devices on a network. NTP uses an NTP server that maintains a highly accurate time. An NTP network is comprised of devices (clients) that are to be synchronized with the NTP server that has UTC time and provides it to the client devices.

Prerequisites

- Install and configure an NTP server on your network and remember the IP address of the NTP server.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

17.1.2. NTP Configuration

Configuration Syntax and Commands

The following sections describe the NTP configuration syntax and commands.

Specifying the NTP Server

To configure the NTP server, you must specify details such as the IPv4 address and the domain name of the NTP server.

Syntax

set system ntp server <server-id> <option>

Attribute	Description
server <server-id>	Specifies the name of the NTP server.
ipv4-address <ipv4-address>	Specifies the IPv4 address of the NTP server.
domain-name <domain-name>	Specifies the domain name of the NTP server.
minpoll	Specifies the minimum interval between requests in seconds. NTP dynamically selects the optimal poll interval between the values specified for minpoll and maxpoll.
maxpoll	Specifies the maximum interval between requests in seconds. NTP dynamically selects the optimal poll interval between the values specified for minpoll and maxpoll.

Example:

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "ntp": {
        "server": [
          {
            "server-name": "NtpServerIPv4",
            "ipv4-address": "192.168.16.1",
            "minpoll": 3,
            "maxpoll": 17
          }
        ]
      }
    }
  }
}
```

NTP Service Configuration

You can enable the NTP service under a specific VRF where NTP server is reachable (usually, the inband management VRF).

Syntax:

set inband-management instance <name> **ntp**

Attribute	Description
<name> <instance-name>	Name of the instance
true/false	Enable or disable Network Time Protocol.

Example: NTP Configuration

```
{
    "rtbrick-config:inband-management": {
      "instance": [
        {
          "name": "mgmt-vrf",
          "ntp": "true"
        }
      ]
    }
  }
```

Setting the System Date & Time Using DHCP Option 42 NTP Servers Information

The RtBrick ONL installer image performs a boot time ntpdate operation to set the system date and time.

By default, it uses the pool.ntp.org but DHCP bindings received over the ma1 interface will override it if DHCP option 42 contains a list of one or more NTP servers.

Separately from this boot time setting, the system will continue to use NTPd to maintain accurate system date and time using the list of NTP servers contained in the RBFS configuration.

The /etc/ntp.boot-time-servers.conf file can be used to verify as its contents will be updated with the NTP servers from DHCP. Also, logs are available in /var/log/rtbrick-boot-time-ntpdate.log.

17.1.3. NTP Operational Commands

OSPF Show Commands

NTP Status

Displays the NTP status information.

Syntax:

show ntp status

Example: NTP Status

```
supervisor@rtbrick.net: op> show ntp status
Associd: 0 Status: 0615 leap_none, sync_ntp, 1 event, clock_sync,
Version: "ntpd 4.2.8p12@1.3728-o (1)", Processor: "x86_64",
System: "Linux/5.4.40-OpenNetworkLinux", Leap: 00, Stratum: 2,
Precision: -23, Rootdelay: 9.791, Rootdisp: 28.874, Refid: 193.203.3.171,
Reftime: eae063f2.d94d557e  Thu, Nov 14 2024 11:56:34.848,
Clock: eae06777.af9fc4c5  Thu, Nov 14 2024 12:11:35.686, Peer: 64212,
Tc: 10, Mintc: 3, Offset: 0.039552, Frequency: -12.332, Sys_jitter: 0.195478,
Clk_jitter: 0.175, Clk_wander: 0.012, Tai: 37, Leapsec: 201701010000,
Expire: 202412280000
```

NTP Associations

Displays the information about the various NTP servers configured.

Syntax:

show ntp association <options>

Option	Description
no-resolve	Details without DNS resolution.

```
supervisor@rtbrick.net: op> show ntp association
Flags: * Synchronized, # Almost synchronized, + Selected for possible
synchronization,"
       – Candidate for selection, ~ Statically configured"
Remote              Refid            St  T  When  Poll  Reach    Delay   Offset
Jitter
+194.50.19.117      131.188.3.222     2  u   522  1024    377    4.939    0.162
0.069
*193.203.3.171      .GPS.             1  u   897  1024    377    9.791    0.008
0.066
-51.75.67.47        157.13.205.54     3  u   770  1024    377    4.929   -0.110
0.149
+85.215.189.120     192.53.103.103    2  u   868  1024    377   13.483    0.043
0.181
-167.235.70.245     195.176.26.205    2  u   757  1024    377    0.883    0.158
0.332
-162.159.200.123    10.163.8.4        3  u   642  1024    377    4.160    0.439
0.484
```

Verifying NTP Service on Linux

To get more information about NTP's status, use the ntpq command:

The command displays the NTP servers that the system is synchronized with.

```
supervisor@rtbrick>LEAF01:~ $ sudo ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
198.51.100.5       .XFAC.          16 u    -   64    0    0.000    0.000   0.000
```

17.2. Logging

17.2.1. RBFS Logging

This document provides you information about logging in the entire RBFS ecosystem. It includes the RBFS container and the host OS which is ONL in case of hardware switches.

RBFS logging involves recording log messages during the execution of events. It generates reports about activities within the entire RBFS ecosystem across various functional areas. Almost every component in RBFS ecosystem can generate log files. Logging enables you to trace application errors in real time.

Understanding RBFS Logging

RBFS's logging system captures events from various components, including Brick Daemons (BDs), non-BD daemons like Prometheus, and host-level services such as CtrlD and ApiGwD. These logs provide insights into the entire RBFS ecosystem, facilitating real-time monitoring and analysis.

RBFS Logging allows you:

- Enable Logging
- View Logs (at var log file location and by using operational commands)
- Export Logs to an External Server

An RBFS ecosystem contains multiple microservices and these microservices are known as Brick Daemons (BD) and other (non-BD) daemons. CtrlD and ApiGwD daemons are part of ONL, but they do not reside inside of RBFS container. BDS provides in-built infrastructure for logging which can be used by all BDS applications.

You can view all the logs generated and stored in the var log files and using operational commands. All logs can be exported to an external management server where you can view and analyze the real-time data.

Log Level

Log level indicates level of the severity of the events (logs). You can configure logging based on different severity levels. RBFS supports the following log levels, listed in order of priority:

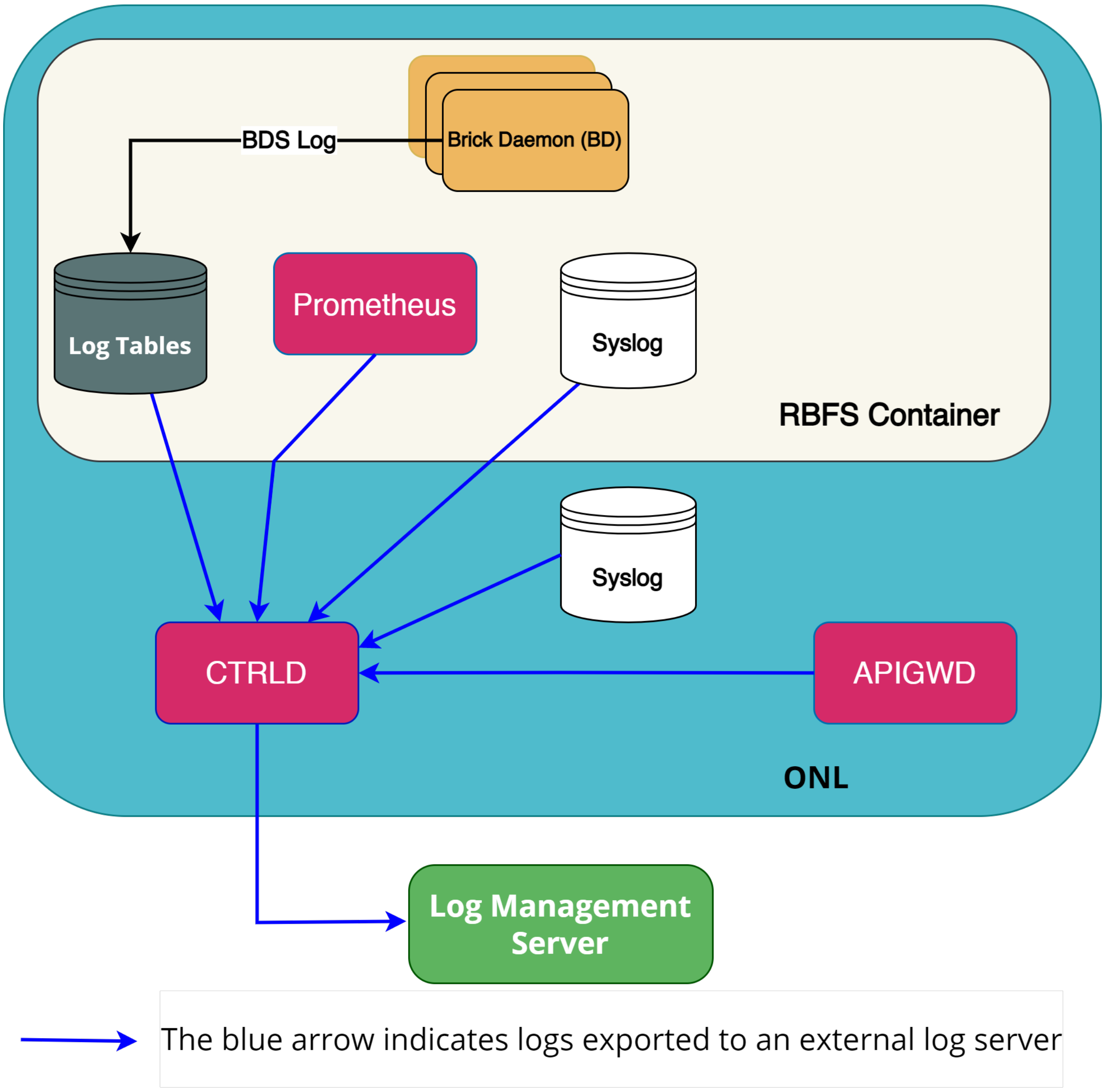
- Emergency
- Alert
- Critical
- Error
- Warning
- Notice
- Info
- Debug
- None

Any level above the level Warning indicates that you should perform logging with caution as a scaled environment may cause a system instability.

About RBFS Logging System

RBFS provides logging for the entire RBFS ecosystem that includes Brick Daemons (BD) and also for other (non-BD) daemons. Brick Daemons are built on top of BDS and other (non-BD) daemons (such as Prometheus) are the ones which are not dependent on BDS.

The RBFS container logging infrastructure provides in-memory (BDS) and traditional (BD) logging support for RBFS applications. The BDS logging is a low-latency and in-memory logging which can be used in a high scale system without compromising much in performance whereas BD logging is a direct write to a file hence CPU-heavy.

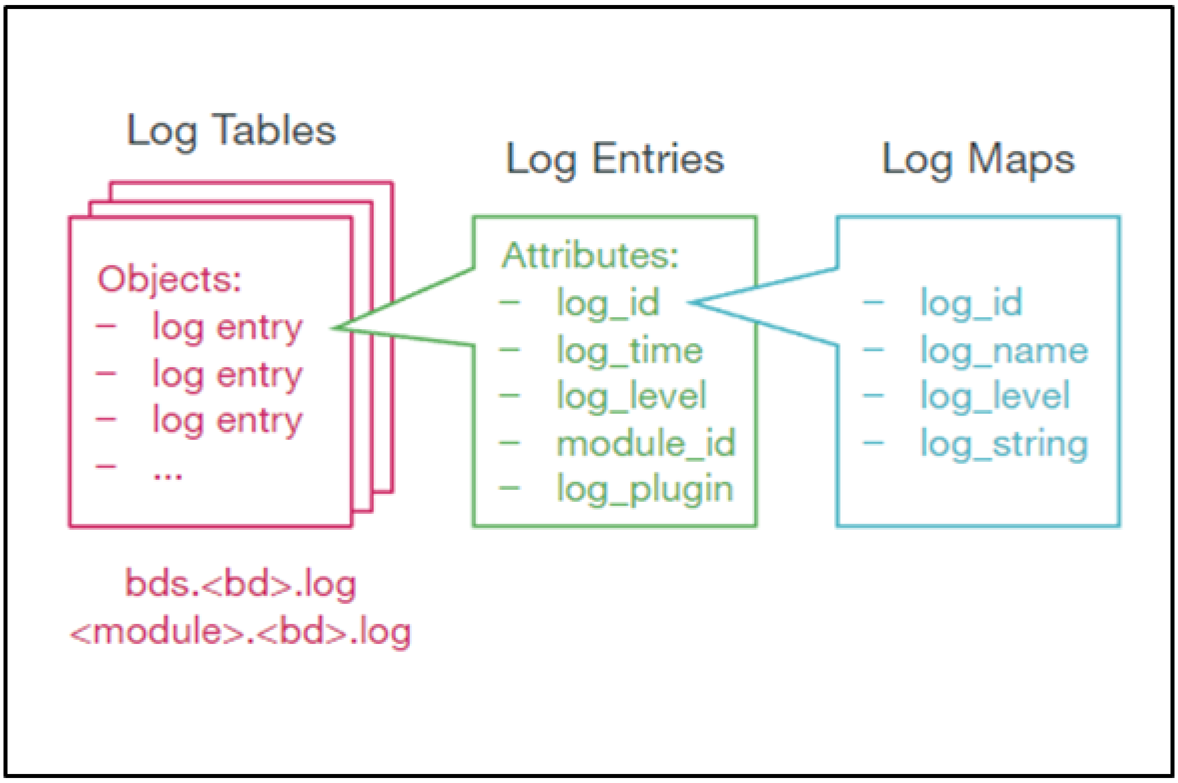


The blue arrow indicates logs exported to an external log server

In RBFS, logs are generated from various components or sources such as BDS, Syslog, CtrlD, APIGwD, and Prometheus. All these logs can be finally exported to a log management server.

BDS Logging

BDS logs are stored in a log table. For every unique event, a log ID is created in RBFS. Whenever that particular event is logged, a log entry gets added to the log table. A log table is created for a module only when that module has at least one event logged. Every module in RBFS has at least one log table named in this format: `<modulename>.<bd-name>.log`.



Log Tables

BDS logs are stored in a BDS table. BDS creates a log table for each module in a BD. One entry is added to this log table for every log. Older entries are removed from the table when the number of entries exceeds more than 10,000.

Log Maps

Every log is mapped to one specific event that is logged by the application. For the optimized usage of memory, RBFS does not store the verbose strings; instead, it stores the log map as an identifier to the actual string message.

Log map and log ID refer to the same entity.

You can access these log maps at the following location:

```
/usr/share/rtbrick/liblog/logs/
```

You can see the log maps, organized based on the modules that they belong to, at:

```
supervisor@rtbrick:/usr/share/rtbrick/liblog/logs$ ls
bds   bgp  fib  fwdinfra  ifm  lldpv2  pd  policy  pubsub  resmon  rib  snapshot
static  time_series
```

In the preceding example, you can see the modules that have registered with the log maps.

If you want to know more about a particular log map, you can perform a *grep* of the log map in this directory.

Log Groups

A log group is a collection of log maps or log IDs. Groups have been introduced to simplify the log configuration tasks. For example, to debug a BGP peer issue, instead of enabling logs for individual log IDs that are related to BGP peer, you can enable log for a log group BGP peer.

Log Modules

Every BDS application consists of multiple modules. Logging can be configured for each BDS modules separately.

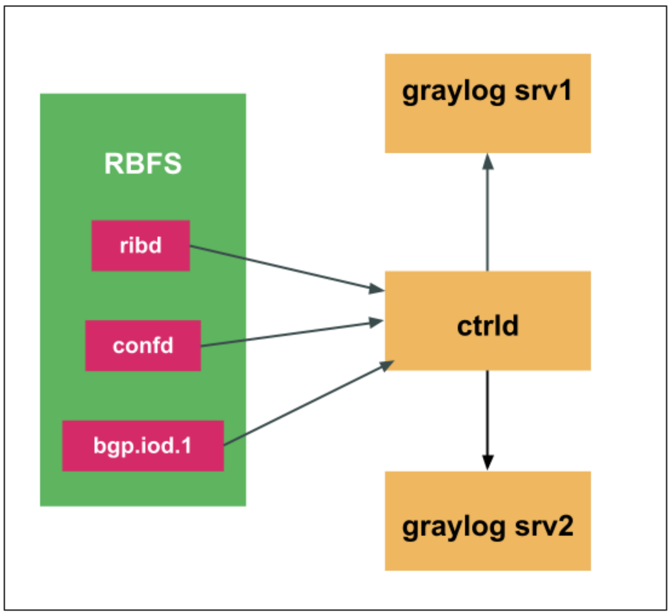
The following are RBFS modules:

```
bcm_q2c
bgp
fib
hostconfd
ifm
igmp
ipoe
isis
l2tp
lag
ldp
license
lldp
mrib
ospf
pim
policy
poold
pppoe
resmon
rib
rtbrick-cli
secure_management
snapshot
subsMgmt
static
time_series
vpp
```

Log Export to an External Server (Plugin Alias)

Any logs in RBFS can be exported to an external logging destination. Currently, CtrlD supports GELF and syslog as external plugins.

CtrlD is the egress node for all the GELF (Graylog Extended Log Format) messages. The brick daemons which are configured to send GELF messages to CtrlD and CtrlD forwards them to the configured endpoints such as syslog or GELF endpoint. This is because CtrlD enhances the GELF message with switch-global settings (for example, the serial number of the switch).

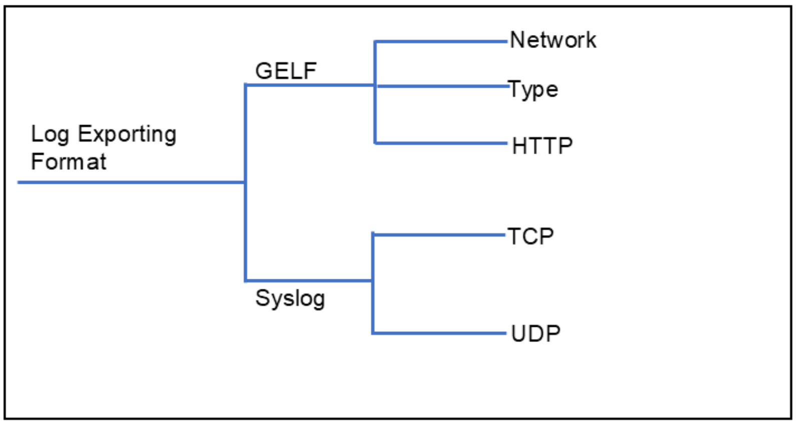


```
supervisor@rtbrick: cfg> set system host log-alias default endpoint
http:10.1.1.1:2002/gelf
  <cr>
  level                     Log level for the alias
  network                   Log alias endpoint port
  queue-size                Queue size that will be used for the fanout [Range: <0-
500>]
  type                      Log alias server type
```

```
supervisor@rtbrick: cfg> set system host log-alias default endpoint
http:10.1.1.1:2002/gelf network
  http                      Log alias endpoint port
  tcp                       Log alias endpoint port
  udp                       Log alias endpoint port
```

Supported Logging Format

RBFS supports two types of logging format: GELF and Syslog.



GELF:

GELF is a logging format for Graylog. It uses JSON to structure log messages in a way that makes them searchable and more structured.

The supported protocol for Graylog is HTTP.

Example:

```
supervisor@rtbrick: cfg> set system host log-alias graylog endpoint
http://10.1.1.1:2002/gelf type
 gelf                 syslog
```

Syslog:

Syslog is a format used by Linux systems and many networking devices for logging. It is a plain text protocol with a defined structure.

The supported transportation protocols for Syslog format are TCP and UDP.

You can configure log-alias to send logs to a centralized logging endpoint, like Graylog, using the GELF.

Example:

```
supervisor@rtbrick: cfg> set system host log-alias graylog endpoint
http://10.1.1.1:2002/gelf network
http
tcp
udp
```

Guidelines and Limitations for BDS Logging

- By default, BDS logging is enabled and the log level is set to 'Error'.
- By default, logging for BDS and PUBSUB modules have been disabled. As these two modules are infrastructure specific, these logs may not be useful for end-users. However, developers can enable logging for these modules using debug commands.
- You can configure log levels per BD, per module, or per group.
- Do not keep logging enabled for longer duration in a scaled setup.
- The following log levels are present in the system. Any level above the level Warning indicates that you should perform logging with caution as a scaled environment may cause a system instability.

If your system becomes unstable, you can remove the logging configuration using the delete log command in configuration mode.

- All log levels lower than the log level specified are logged. For example, if the specified log level is "Warning", then all logs that come before "Warning" (Emergency, Alert, Critical, Error, Warning) are logged.
- When you set the log-level to "None", that means log has been disables for the specific module, group, or global.

Syslog

Syslog is generated by an API based logging mechanism provided by Linux. Some of the open source libraries present in RBFS use Syslog as a logging mechanism.

Host operating system and RBFS Linux container use syslog for logging. Syslog can also be exported to Graylog server.

Syslog messages can also be exported to CtrlD and these messages are forwarded to the defined log management servers. Currently, RBFS supports exporting syslog from the Linux and ONL system facilities such as auth, authpriv, daemon, and kern to Graylog

For information about Syslog configuration, see section System Logging Configuration.

TSDB (Prometheus) Alert Messages

Prometheus is the systems and service monitoring application, which can be deployed in RBFS, to collect and process metrics. In RBFS, alert messages generated from Prometheus are forwarded to CtrlD and these messages, from CtrlD, can be exported to the configured log management servers.

Logging in ONL

In RBFS, there are daemons, which are not part of RBFS container, but run on the ONL host. RBFS provides logging for these daemons. The following are the daemons which reside on the ONL host:

```
hostconfd
CtrlD
ApiGwD
```

CtrlD and ApiGwD Logs

CtrlD logging provide log messages of events related to business, elements, ZTP, and security. ApiGwD logs contain details about who accessed the API and how they accessed it. hostconfd provides rest APIs to interact with container and ONL linux services.

ApiGwD`and `CtrlD send different log messages about status changes or progress of processes to the configured GELF endpoint.

Supported Platforms

Not all features are necessarily supported on each hardware platform. Refer to the *Platform Guide* for the features and the sub-features that are or are not supported by each platform.

17.2.2. Logging Configuration

Configuration Overview

BDS (Brick Data Store) is an in-memory state database and a core component of RBFS. It stores all system state information as structured objects within its tables. Acting as a central communication hub across components, BDS generates a substantial volume of logs that are essential for monitoring.

You can enable logging for:

- Brick Daemon (such as pppod, ipod, confd, and so on)
- Modules (such as OSPF, BGP, ISIS, and so on)
- Host (ONL) system

Once logs are generated, you view logs:

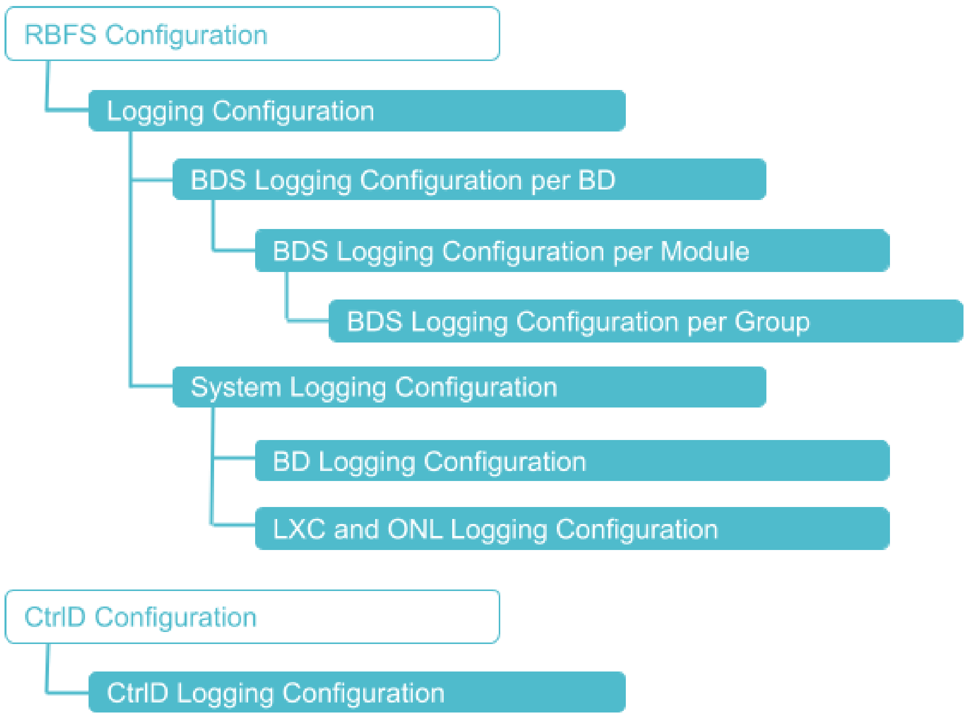
- In the var log files
- Using operational commands

You can export the logs to an external server using the formats:

- GELF
- Syslog

Configuration Hierarchy

The diagram illustrates the logging configuration hierarchy.


RBFS Configuration
Logging Configuration
BDS Logging Configuration per BD
BDS Logging Configuration per Module
BDS Logging Configuration per Group
System Logging Configuration
BD Logging Configuration
LXC and ONL Logging Configuration
CtrlD Configuration
CtrlD Logging Configuration

Configuration Syntax and Commands

The following sections describe the logging configuration syntax and commands.

BDS Logging Configuration

You can configure BDS logging for a BD, a module, and for a group.

A specific configuration takes priority over a generic configuration. For example, if you have configured a global log level of bgp.iod.1 to "warning", and you have configured a log level of bgp module to "notice", then the final log level of bgp will be "notice".

Configuring BDS Logging for a Brick Daemon

You can configure BDS logging for a brick daemon or all daemons to generate logs related that daemon/daemons.

Syntax:

set log bd <bd-name> <option> …

Attribute	Description
<bd-name>	Configure for the specified BD name.
all	Configure for all BDs.
module <module-name>	Module name. For more information, see the section "Configuring BDS Logging for a Module".
plugin-alias alias-name <alias-name>	Specify the external log management server name for sending the logs to the server.

Attribute	Description
plugin-alias level <level>	Log severity level. You can filter logs based on the log severity levels for sending to the external log management server. This allows to send only the required log messages to the log management server instead of sending a whole lot of data. For example, if you configure Warning as the severity level, logs with the severity level 'Warning' and all the log levels above Warning such as Error, Critical, Alert, and Emergency will be sent to the log management server.

Example 1: BDS logging for a BD Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {},
    "rtbrick-config:log": {
      "bd": [
        {
          "bd-name": "bgp.iod.1",
          "level": "info",
          "plugin-alias": {
            "alias-name": "ztp",
            "level": "error"
          }
        }
      ]
    }
```

Configuring BDS Logging for a Module

You can enable logging for a module such as BGP, IS-IS, and so on.

Syntax:

set log module <module-name> <option> ...

Attribute	Description
<module-name>	Module name

Attribute	Description
group <group-name>	BDS log module log-group configuration. For more information, see the section "Configuring BDS Logging for a Group".
level <level>	Log severity level.
plugin-alias alias-name <alias-name>	Plugin alias name
plugin-alias level <level>	Log level. You can filter logs based on the log severity levels for sending to the log management server. This will help you to send only the required log messages to the log management server instead of sending a whole lot of data. For example, if you configure Warning as the severity level, logs with the severity level 'Warning' and all the log levels above Warning such as Error, Critical, Alert, and Emergency will be sent to the log management server.

Example 1: Logging Module Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {},
    "rtbrick-config:log": {
      "module": [
        {
          "module-name": "igmp",
          "level": "info",
          "plugin-alias": {
            "alias-name": "graylog-srv1",
            "level": "warning"
          }
        }
      ]
    }
  }
}
```

Configuring BDS Logging for a Group

You can enable logging for a group of log IDs. This configuration allows you to generate logs for a group of log IDs (also known as log maps). Using log groups, you can enable logging for a specific set of log IDs. This approach gives you more targeted control over log generation, helping you collect only the relevant information without overwhelming the system.

Syntax:

set log module <module-name> **group** <group-name> <option> ...

Attribute	Description
<group-name>	Group name
level <level>	Specifies the level of the plug-in alias.
rate-limit <rate-limit>	Rate-limiting is only supported for log groups. Configuring a higher rate-limit for a whole module may cause system instability due to generation of high volume of logs. The default value is 10.
plugin-alias alias-name <alias-name>	Plugin alias name
plugin-alias level <level>	Specifies the level of the plug-in alias. You can filter logs based on the log severity levels for sending to the log management server. This will help you to send only the required log messages to the log management server instead of sending a whole lot of data. For example, if you configure Warning as the severity level, logs with the severity level 'Warning' and all the log levels above Warning such as Error, Critical, Alert, and Emergency will be sent to the log management server.

Example 1: Logging Group Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:log": {
      "module": [
        {
          "module-name": "bgp",
          "group": [
            {
              "group-name": "interface",
              "level": "warning"
            }
          ]
        }
      ]
    }
  }
}
```

System Logging Configuration

LXC and ONL Logging Configuration

In addition to configuring logging for the various daemons and modules in RBFS container, you can also enable logging for the underlying ONL host syste. You can send logs from Linux and ONL system facilities such as auth, authpriv, daemon, and kern to the log management server.

Also, you can configure an external log management server to transport logs. Currently, Syslog and Graylog are the log management servers supported by RBFS.

Syntax:

set log system facility <facility-name> **plugin-alias** <attribute> <value>

Attribute	Description
<facility-name>	Supported facilities include all, auth, authpriv, daemon, and kern.
plugin-alias alias-name <alias-name>	Plugin alias name
plugin-alias level <level>	Specify the level of the plug-in alias. You can filter logs based on the log severity levels for sending to the log management server. This will help you to send only the required log messages to the log management server instead of sending a whole lot of data. For example, if you configure Warning as the severity level, logs with the severity level 'Warning' and all the log levels above Warning such as Error, Critical, Alert, and Emergency will be sent to the log management server.

Example 1: System Log Configuration

```
{
  "ietf-restconf:data": {
    "rtbrick-config:log": {
      "system": {
        "facility": [
          {
            "facility-name": "kern",
            "plugin-alias": {
              "alias-name": "graylog",
              "level": "notice"
            }
          }
        ]
      }
    }
  }
}
```

External Log Server (Plugin Alias) Configuration

You can configure an external log management server as a destination to send logs.

Syntax:

set system host log-alias <attribute> <value>

Attribute	**Description**
<name>	Log alias name
<address>	Plugin address. For example, Gelf: http://11.1.1.1:1102/gelf Syslog: 10.1.1.1:8008
level	Severity level for the log.
network	Log alias endpoint port.
type	Server type of log alias.

Example 1: Log alias Configuration

```
set system host log-alias bng endpoint http://10.1.1.1:1102/gelf
set system host log-alias bng endpoint http://10.1.1.1:1102/gelf type gelf
set system host log-alias bng endpoint http://10.1.1.1:1102/gelf network http
set system host log-alias bng endpoint http://10.1.1.1:1102/gelf level info
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "host": {
        "log-alias": [
          {
            "name": "bng",
            "endpoint": [
              {
                "address": "http://10.1.1.1:1102/gelf",
                "type": "gelf",
                "network": "http",
                "level": "info"
              }
            ]
          }
        ]
      }
    }
  }
}
```

This following configuration sets up how system logs will be exported to an external log collector, in this case, Graylog.

Example configuration for Graylog with the domain name:

```
set system
set system host
set system host log-alias graylog
set system host log-alias graylog endpoint http://10.1.1.1:2002/gelf
set system host log-alias graylog endpoint http://10.1.1.1:2002/gelf type gelf
set system host log-alias graylog endpoint http://10.1.1.1:2002/gelf network http
set system host log-alias graylog endpoint http://10.1.1.1:2002/gelf level info
```

```
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "host": {
        "log-alias": [
          {
            "name": "graylog",
            "endpoint": [
              {
                "address": "http://10.1.1.1:2002/gelf",
                "type": "gelf",
                "network": "http",
                "level": "info"
              }
            ]
          }
        ]
      }
    }
  }
}
```

Example for Syslog configuration with TCP:

```
set system
set system host
set system host log-alias syslog
set system host log-alias syslog endpoint 10.200.32.49:525
set system host log-alias syslog endpoint 10.200.32.49:525 type syslog
set system host log-alias syslog endpoint 10.200.32.49:525 network tcp
```

```
supervisor@rtbrick: cfg> show config
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "host": {
        "log-alias": [
          {
            "name": "syslog",
            "endpoint": [
              {
                "address": "10.200.32.49:525",
                "type": "syslog",
                "network": "tcp"
              }
            ]
          }
        ]
      }
    }
  }
}
```

Example for Syslog configuration with UDP:

```
set system
set system host
set system host log-alias syslog
set system host log-alias syslog endpoint 10.200.32.49:525
set system host log-alias syslog endpoint 10.200.32.49:525 type syslog
set system host log-alias syslog endpoint 10.200.32.49:525 network udp
```

```
supervisor@rtbrick: cfg> show config
{
  "ietf-restconf:data": {
    "rtbrick-config:system": {
      "host": {
        "log-alias": [
          {
            "name": "syslog",
            "endpoint": [
              {
                "address": "10.200.32.49:525",
                "type": "syslog",
                "network": "udp"
              }
            ]
          }
        ]
      }
    }
  }
}
```